

Ddos Attack Detection Using Neural Network Based On Sdn

Research Article

Volume 6 Issue 1- 2025

Author Details

Asia Othman Aljahdali*, arah AlMarhabi, Nuha Alsaadi,
Cybersecurity Department, College of Computer Sciences and Engineering, University of Jeddah, Saudi Arabia

*Corresponding author

Dr. Asia Othman Aljahdali, University of Jeddah, Cybersecurity Department, College of Computer Sciences and Engineering, Saudi Arabia

Article History

Received: April 11, 2025 Accepted: April 17, 2025 Published: April 23, 2025

Abstract

Among the deadliest threats, today are cyber-attacks. DDoS (Distributed Denial of Service) is one of them. A cyberattack occurs when an attacker temporarily or permanently shuts down a network or machine linked to the network. During an attack, excessive requests are sent to the target system to overload it, thereby rendering it unusable. We present a method for detecting DDoS attacks based on neural networks. Malicious and legitimate data flows would be flagged, preventing network degradation. Existing models in the field were compared with the proposed system. Results showed that our model was 99.74% accurate and more accurate than existing models.

Introduction

DDoS attacks are one of the most common types of cyberattacks. It makes resources and services unavailable to users. As a result of a DDoS attack on a network, legitimate users will no longer be able to utilize the service or resources anymore. Several methods exist for detecting DDoS attacks in networks, including machine learning, hybrid methods, and statistical methods.

DDoS attacks can be launched for a few reasons, including political, financial, and personal gains. This problem has been exacerbated by an increasing reliance on the internet and data-driven applications, along with cheap online stress boosters. DDoS attack detection and mitigation have become a top priority due to the rising number of successful attacks as well as the variety of types of DDoS attacks. Federal Bureau of Investigation (FBI) officials announced that stress services should not be used to conduct a DDoS attack. Arbor Networks Inc. also reported a 600Gbps-scale attack in 2017. According to the research, 63% of data center operators expressed interest in deploying Software Defined Networking (SDN) technologies to detect and mitigate DDoS attacks. To address these cyber security concerns, we need a new network paradigm. A common factor of DDoS attacks is abnormal traffic sent to the victim. This pattern can be easily detected when compared with network operation parameters before an attack. Pattern recognition can be used to identify DDoS attacks by various detection

mechanisms. The type of detection mechanism to be used depends on network characteristics such as protocols, CPU utilization, delay, throughput, packet header, and packet size.

A DDoS attack is considered to occur when access to network resources such as a server or a computer is intentionally blocked, or performance is degraded due to malicious action taken by another user. However, the availability of the victim's resources may be compromised, even though data may not be damaged directly or permanently. DDoS attacks are not a new threat, and there is a lot of research on the detection and mitigation of these attacks. As a result of similarities between legitimate and malicious packets, it continues to thrive and evade detection. With the adoption of Software-Defined Networking (SDN), Network Function Virtualization (NFV), and cloud computing, more needs to be done regarding intrusion detection and defense mechanisms.

The concept of machine learning has been used in many fields lately to determine or predict patterns of incidents. This approach has been used in detecting and mitigating DDoS attacks. It involves training datasets and comparing them with a real-time data stream of the same attributes to categorize attacks as benign or malicious. [1] presents a review of how to handle DDoS attacks in SDN using several machine learning techniques that can be deployed to detect and mitigate DDoS attacks in SDN with associated merits and demerits identified.



Furthermore, the study by [2] approves that SVM is a better choice than artificial neural network (ANN) and naive Bayes in terms of accuracy and recall value [2], but it has less precision [3]. The proposed ML approaches by [3] are not substantiated by different topologies. Currently, there isn't a single robust mechanism for detecting and mitigating DDoS attacks. Recent research focuses on effectively combining one or two approaches to combat DDoS attacks in SDN. Additionally, a statistical approach has been used to extract traffic flow features while the ML approach is applied to classify network anomalies. Adaptive parameters of detection algorithms are optimized by deploying a testbed to evaluate the approach in real time and on real devices in an adaptive manner.

We propose an adaptive mechanism to improve detection accuracy. The main objective of this study is to detect DDoS attacks in the SDN environment by Implementing a neural network model to maintain a suitable accuracy rate. Additionally, the study evaluates the proposed solution by measuring and comparing the effectiveness of the detection rate and accuracy of the proposed solution to the previously available results. The aim of this research is to mitigate and prevent DDoS attacks on the network while keeping the system and resources running and available for legitimate users. Machine learning has been integrated with the SDN controller to detect and drop attack traffic while virtually no innocent traffic is affected. A new algorithm based on these features has been developed to recognize and drop attack traffic.

Background

DDoS Attack

According to the Cisco cybersecurity report, 30 % of organizations reported cyberattacks in 2019 due to internet cybercrime. According to [4], the most common attacks on the Internet are denial of service (DoS) and distributed dos (DDoS). During DDoS attacks, attackers use botnets to control many hosts, sending huge requests to victims to disrupt their services. This results in the destination victims' limited resources being overburdened and unable to provide service to legitimate users.

Studies documenting in-depth discussions regarding the detection of DDoS attacks [5,6]. Based on the categorization of DDoS attacks, several of them can be classified into three groups, as shown in Figure 1. An attacker who uses volume-based attacks consumes all internet bandwidth between the target and the attacker. The amount of traffic sent to a target is massive because amplification methods or other methods create a large amount of traffic, such as requests from botnets. This type of attack is well illustrated by the ICMP flood attack. In protocol-based attacks, also called state-exhaustion attacks, the state table capacity of web servers or intermediate resources, such as firewalls or load balancers, is consumed to disrupt services. In protocol attacks, the target is rendered unreachable by exploiting weaknesses at layer three and layer four of the protocol stack, such as the TCP-SYN flood.

Applications layer attacks aim to exhaust a target's resources. An attacker connects to the target and then monopolizes processes and transactions on the server. They do this by exploiting a weakness in the layer 7 protocol stack [5,6] Volume- and protocol-based attacks send a large amount of traffic, making them easier to detect than application-based attacks. We study flood attacks in this article, although there are many types of DDoS attacks, and suggest a novel way to detect them. TCP-SYN flood attacks exploit a weakness in the TCP protocol stack. The three-way initiates a TCP connection handshake, which is vulnerable. Kaspersky reports that more than 50% of DDoS attacks in Q4 2020 were associated with TCP-SYN flooding. In contrast, an ICMP flood consists of sending large numbers of unacknowledged, malicious ICMP messages, thus exhausting the vulnerability's resources. TCP-SYN flooding is the most common type of DDoS attack in Q4 2020, accounting for more than 50% of all attacks. By overloading the network bandwidth with ICMP echo-request packets, the attacker makes the target device inaccessible to normal traffic [5,7]. Several approaches are used as detection techniques for such attacks, including simulation, offline analysis, and complex operations. It is difficult to evaluate the performance of the detection and mitigation algorithms in real time and context.

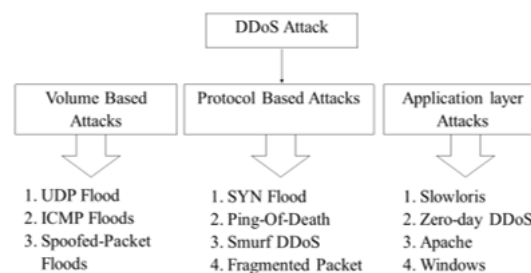


Figure 1: Distributed denial-of-service (DDoS) attacks classification.

Software-Defined Networking (SDN)

Due to the development of big data, cloud computing, and other emerging technologies, network traffic is constantly growing, and traditional network architecture with IP at its core struggles to meet the demands of network scalability, management, and adaptability. Software-defined networks (SDNs) are characterized by the separation of the control plane and data plane, the centralization of the network state, and the isolation of the controller from its underlying network devices. SDN enhances network management, extensibility, controllability, and dynamics. In recent years, as SDN applications have grown in popularity, SDN security has become one of the most important study areas. Its high destructive power, ease of implementation, and lack of simple and realistic defenses make distributed denial-of-service (DDoS) attacks one of the most significant internet

security threats today. The attacker builds a puppet and sends an organized attack to the expand bandwidth, CPU capacity, memory, and other assets of the attacked targets [8]. Conventional defenses against DDoS assaults focus on activity cleaning and security system blocking, making it difficult to realize coordinated planning of the entire arrangement and leading to expensive resource overhead. While SDN provides a modern opportunity for locating DDoS attacks since it enables real-time observation of the full network as well as the activity state of every node with its centralized control and programmability.

Software-Defined Networking (SDN) separates the control plane from the data plane. Even though the forwarding engine is in the switch, a centralized software controller controls all network control functions, such as traffic monitoring and routing. The SDN architecture becomes flexible by enabling network administrators to add sec-

urity functions to the controller using programming languages such as Python, Java, or others. Recent research has focused on the deployment of SDN in network security [9]. The SDN architecture contains three layers:

a. The SDN Data Plane comprises a group of individual or more network components, like a tangible and a virtual switch. A group of network switches as well as routers in the data centre. The SDN networking schemes control the transmitting and information processing efficiencies of the structure. This involves addressing and handling the data pathway. Resources are continually an abstraction of fundamental tangible efficiencies. Accordingly, switches transmit packets in a data plane layer.

b. SDN controller plane: It consists of a group of SDN controllers that include restricted control over a set of resources demonstrated in the data plane. It is a system that receives commands or conditions from the application layer and relays them to the networking layer. From the hardware tools, the controller extracts instructions about the network. This allows it to communicate back to SDN applications about a conceptual aspect of the network, containing statistics and performance information. One of the SDN controller's actions is to re-optimize the capability distribution. responds to network occurrences to restore the network. serves as a control component in a response loop.

c. SDN Application Plane: These applications collaborate with SDN controllers and schemes via APIs. By leveraging all network information around network state, topology, etc., the application tier is an open field for developers to develop as many creative applications as possible. By gathering information from the controller for making-decision intentions, this application can form an abstract concept of the network. There are many types of advanced applications, including commercial applications, network configuration and administration, network industrialization, network control, network mediation, network procedures, and network protection. Many SDN applications are capable of supporting a wide range of enterprise and data center solutions. See Figure 2.

As shown in Figure 3, the SDN controller is the essential element of the control plane, as it is the "intelligence" of the network and allows interfaces to more planes. The communication accompanying the application plane is created over the northbound connection, a set of open APIs (Application Programming Interfaces) that clarifies the action of establishing a network application. Communication with the data plane is over the southbound connection that authorizes connection between the SDN controller and the switches. The most popular protocol utilized in the southbound connection is OpenFlow, a substantial protocol in the SDN environment, preserved separately by the Open Network Foundation and through all important network device vendors.

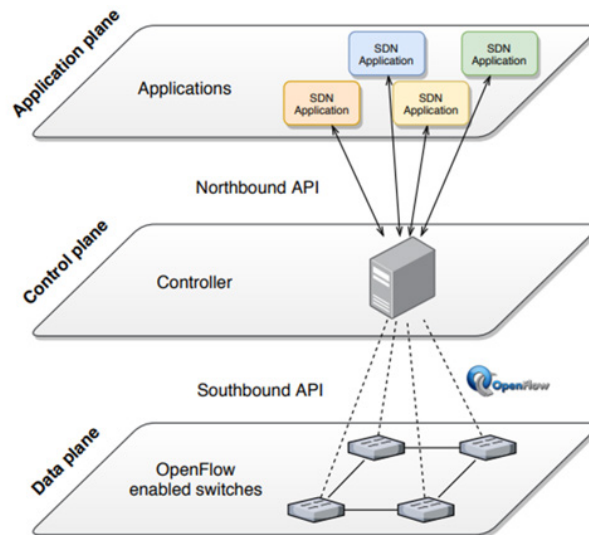


Figure 2: Software Defined Network (SDN) architecture.

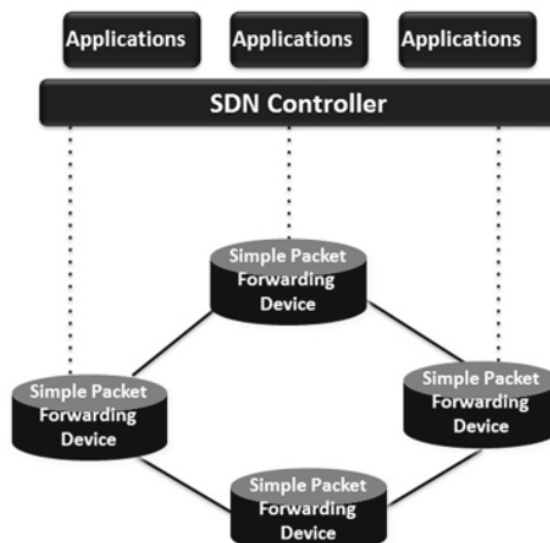


Figure 3: Software-defined networks (SDN).

Currently, a new way called Stateful SDN has comprehensive the fundamental functionalities of OpenFlow and involves the competencies to request various match-action standards established in various states of the SDN flow tables of the switch. With these capabilities, the switch executes the capability of responding to packet-level occurrences and generating capacities on the fly. If the outcome of the evaluation equals the standards that the switch has in the flow tables, then action is taken according to the standard. The advancement of new applications and services is enhanced as the control of the network is sufficiently delivered, and adding a new characteristic is made easier by deploying new applications in the controller. states of the SDN flow tables of the switch. With these capabilities, the switch executes the capability of responding to packet-level occurrences and generating capacities on the fly. If the outcome of the evaluation equals the standards that the switch has in the flow tables, then action is taken according to the standard. The advancement of new applications and services is enhanced as the control of the network is sufficiently delivered, and adding a new characteristic is made easier by deploying new applications in the controller.

The SDN architecture consists of diverse types of elements (Network Entities (NE), northbound and southbound interfaces, SDN controllers, and SDN requests). While traditional networks only contained individual network tools, Today, not only are essential network tools secured, but likewise applications, controllers, and the systems of connections among them. In Table 4, you can see the primary attacks in which SDN deliberately exposed itself and the protection feature that was compromised.

SDN applications use APIs from the controller to perform functions that are used in the northbound connection outlined as an attack heading. Northbound APIs implements automation and programming languages. Although a malicious attacker attacks the exposures of such programming languages or technologies, the user can accept commands of the SDN network by way of the controller. Furthermore, if an exposed northbound API is abused, the attacker is wise to develop their SDN approaches and thus gain control of the SDN environment. The southbound connection may be more of an attack heading on account of the many southbound APIs and protocols utilized by the controller to correspond with the network elements. The essential APIs are OpenFlow, Secure Shell (SSH), Simple Network Management Protocol (SNMP), or Border Gateway Protocol (BGP), with Chao ice. Each of these protocols has its mechanisms for protecting the connection to network components. However, if a malicious attacker attacks their exposures, the flows are reduced or formed into the equipment's flow table, authorizing illegitimate traffic or changeable routes to enforce a Man in the Middle Attack (MITM).

DDoS Detection Techniques

Several techniques can be used to detect and mitigate DDoS attacks, including statistical tarnishing, hybridization, and machine learning. The existing techniques run from conventional measurable techniques and advanced machine learning calculations to the combination of numerous techniques, and after that to complex profound learning techniques, all taking great advantage of the worldwide view and centralized oversight capacity of the control plane to move forward the accuracy of DDoS assault discovery. In any case, due to stream collection, measurements, classification, and so on is all that is needed to prepare an SDN controller, when the organized scale is expanding, the controller must confront tremendous overhead, coming about in attack discovery delay. And the worst part is that since the DDoS attack was discovered, the controller has been overburdened, or even down. At the same time, the controller must regularly get stream table and bundle data from the edge switch for assault discovery. When the organized scale increments, the burden on the southbound interface will be overwhelming. Subsequently, within the handle of DDoS attack location, how to diminish the burden of the controller and the south-

bound interface as well as increase the attack detection speed while guaranteeing the discovery precision is a vital research subject. For this reason, we use a neural network algorithm to maintain a suitable accuracy rate to detect DDoS attacks using SDN.

Statistics-based solutions analyze traffic between the standard and attack phases using statistical techniques. [9] reports a method based on deviation from the normal distribution of throughput when the server is in a normal state. By detecting deviations from the average throughput value, an attack can be detected. Sufian Hameed et al. present a collaborative DDoS mitigation protocol that permits SDN controllers in different autonomous systems to securely communicate and share attack information. According to [8], the authors developed an SDN-based DDoS attack detection framework that performs a two-stage granularity filtering procedure between the coarse-grained detection data plane and the fine-grained detection control plane. Switches and controllers are SDN-enabled. A lightweight flow monitoring algorithm is used to analyze and detect DDoS attacks in SDN-enabled switches. An article [8] proposes using SDN's flow monitoring capability to detect DDoS attacks better. By utilizing measurement resources available throughout the SDN network, an adaptive balance between coverage and attack detection particle size is achieved.

The machine learning-based solution uses smart algorithms to discover hidden features inside network traffic in normal and attack states. K-Nearest-Neighbor (KNN) is used to detect ICMP attacks. KNN is used to assign network status labels based on the k nearest neighbors' labels. Other machine-learning detectors Methods are based on determining the time of request between hosts [10], which is done by obtaining the request time, several source hosts, and the number of the destination host, and then classifying traffic as normal or attack using various algorithms (Naive Bayes, KNN, K-means). Using the entropy method to determine the randomness of the flow data, Ref. [11] presents a novel solution for the early detection and mitigation of TCP SYN flooding. The entropy information includes destination IP and a few attributes of TCP flags. It is implemented as a Floodlight extension module and evaluated under different conditional scenarios. [4] propose an efficient and lightweight framework to detect and mitigate DDoS attacks in SDN using features' entropy and an SVM algorithm. Firstly, the network traffic information is collected through the SDN controller and slow agents. Then an entropy-based method is used to measure network features, and the SVM classifier is applied to identify network anomalies. Another approach using the SVM algorithm is presented by [12]. By integrating SVM into the Ryu controller, the paper focuses on anomaly traffic detection based on the entropy of IP source addresses and ports. The system's performance reliability Mininet can hardly be evaluated in real-time.

Even though recent research has focused on detecting DDoS attacks, it is not easy to deploy a system that can handle massive traffic in a short period. In addition, it is challenging to distinguish innocent traffic from attack traffic. DDoS attacks are mitigated by dropping all packets to protect the target. Normal traffic is also blocked. [11] proposes an alternative solution based on SDN and an Intrusion Detection System (IDS) to mitigate network attacks. The authors suggest interfacing Snort IDS [12] with a Tonka (2021) controller by using a Unix Domain Socket. Snort can then alert the SDN controller to enforce policies on SDN-enabled switches to block malicious IP addresses listed in block lists. Snort can detect a variety of attacks, but there are some drawbacks. Rather than using machine learning algorithms to adapt policies to the actual situation of traffic at the point of deployment in a specific network, it deploys Snort's predefined rules. In addition, MININET is used for performing the tests, which cannot guarantee real-time performance. Our next section explores malicious TCP-SYN and ICMP flood attacks on an ISP network. In this work, machine learning algorithms are used to classify traffic into the attack and expected, so that attack traffic can be mitigated. In addition to the mitigation mechanism, a real-world testbed with real network devices



and servers has also been implemented, enabling a real-time evaluation of the newly developed system under current network conditions.

SDN-Based DDoS Attack Detection Techniques

To detect and mitigate DDoS attacks, SDN provides many unique features. These features include the separation of the control plane from the data plane, a logically centralized controller, the programmability of the network by external applications, software-based traffic analysis, and the capability to update forwarding rules dynamically. DDoS attack detection and mitigation techniques using SDN can be done using several approaches, including Entropy, machine learning, and traffic analysis [12]. Entropy-based methods for detecting anomalous network activity require evaluating the distribution of network features. To calculate entropy, probability distributions of various network features such as source IP addresses, destination IP addresses, and port numbers are utilized. Anomalies are identified by measuring changes in entropy values based on predefined thresholds. Anomalies can be detected using machine learning algorithms such as Bayesian networks and fuzzy logic. To detect anomalies, these algorithms use various network features and traffic characteristics. Traffic pattern analysis assumes that infected hosts exhibit similar behavior patterns to those of benign hosts. Typically, a botnet attack involves the control of infected machines (bots) by a single botmaster. The same traffic patterns are observed due to the same command being sent to many botnet members (e.g., sending illegitimate packets, starting to scan). [7] The rate of connection techniques is divided into two categories:

I. connection success ratio and

II. connection rate,

with connection rate referring to the number of connections created within a given period. OpenFlow and SNORT integrated are used to detect attacks and reconfigure the network dynamically, these techniques use a combination of intrusion detection systems (such as SNORT) and OpenFlow. Intrusion detection systems monitor traffic to identify malicious activity. In response, OpenFlow switches are dynamically reconfigured in real-time according to detected attacks

Related Work

[14] use machine learning in SDN-based ISP (Internet Service Provider) networks to mitigate DDoS attacks, especially TCP-SYN floods and ICMP floods. As a result of rapid and inconsequential neural network innovation, K-Nearest-Neighbor (KNN) is standardized for performing real-time detecting and treating attacks by seeking back to the IP attack beginning, while sane traffic is barely affected. Furthermore, they proposed a machine learning method for automatically fitting scanning window period traffic input for better mitigation effectiveness. It has been analyzed in real-time how their suggested algorithms work, and the experimental results accompanied by the CAIDA traffic traces, in addition to the botnet traffic from the testbed, indicate that over 98% of the attack, traffic is discovered and dropped. Detecting low-rate distributed denial of service using the hidden Markov model (HMM-R) algorithm [15] (L-DDoS) attacks in the data center network. They utilized SDN mechanisms to carry out intelligent control and set of traffic, and PACKET-IN communication was used to set the discovery time to defeat the discovery period. Then they made the Renya Entropy as a probability assigned to lower the false positive in the feature. Finally, they utilized HMM-R to determine the sort of state to discover L-DDoS attacks at various rates in the form of possibility. They distinguished HMM-R from other algorithms such as K-Nearest-Neighbor (KNN), support vector machine (SVM), self-organizing map (SOM), and backpropagation (BP). They showed that the HMM-R can raise the true positive and decrease the false positive rate at various attack rates. It has an inclusive detection efficiency. Exceptionally, at a reduced rate, the performance of the HMM-R algo-

rithm increases considerably.

Using SDN, Tan et al. introduce a new scheme for the defense and detection of DDoS attacks that efficiently advances the accuracy and effectiveness of discovery and prevents attacks on SDN. To screen for unusual flows in the network using a trigger device for DDoS attack detection in the data plane. As a result, they use K-Means and KNN mixed machine learning algorithms to detect distrustful flows by employing asymmetrical and rate characteristics of the flow. Afterward, the controller will take appropriate measures to defend against the attacks. They resolve the detection and defense technique of DDoS attacks over SDN which integrates SDN's improvements and machine learning algorithms and adopts a more focused method to detect and prevent DDoS attacks in the SDN controller. Their experiments are built to confirm that the detection method suggested in this thesis achieves good results. Additionally, the detection fulfill method can efficiently detect the occurrence of unusual flows and preserve the resources of the controller. The selected defense method can further adequately mitigate DDoS attacks.

[16]. They planned a DDoS assault discovery framework stage considering the open-source Floodlight regulator in SDN. In the assault recognition trigger module, we propose a discovery trigger instrument given the PACKET_IN message to altogether diminish the reaction time to the assault and the burden on the regulator. In the stream table passage assortment module, we join the components of the OpenFlow convention and DDoS assault to plan a stream table element-based DDoS assault discovery strategy, otherwise called a stream table discovery strategy. By acquiring the stream table sections in the OpenFlow switch, after factual examination, the comparing stream highlights are extricated, and the overt repetitiveness is included in the mix with the component choice calculation under various convention types. In the assault identification module, an order calculation is utilized to prepare the examples and fabricate a recognition model to decide if there is a DDoS assault in the organization. The creator confirms the adequacy and benefits of the framework through tests.

[Manso et al., 2019] has introduced a structure consisting of an IDS that automatically detects various DDoS attacks; therefore, when an attack is discovered, it informs SDN's controller. Their framework detects DDoS attack schemes and reduces them to normal on the user side. This method alleviates the negative effects of the extensive impact of the attacker on probable victims. They outline more downloads and some appropriate traffic-sending outcomes from the SDN controller to network tools. The proposal's results imply that their suggestion conveniently detects several types of cyber-attacks based on DDoS, mitigates the attack's negative effects on network efficiency, and provides the correct data delivery for usual traffic. Their framework focuses on programming relevance over an abstracted concept of the network framework to conveniently detect botnet exploitation, mitigate abnormal traffic at its beginning, and preserve advantageous traffic. [4] performed a network intrusion detection system (NIDS) integrated with an SDN-located information security defense mechanism (ISDM) to conduct abnormality reduction, detection, and defeat damage caused by a DDoS attack.

The analysis outcomes demonstrate that SDN controllers can evaluate suspicious IP addresses linked with domain names by recommending a blocked list recommended by the evaluation results from ISMD. The exploratory results prove that the SDN controller authorizes a defender to react to locate insecure threats and expand mitigation methods for DDoS attacks by utilizing behavioral investigation accompanying logs gathered from OpenFlow switches. Furthermore, the real-time restoration of detection rules based on attack signatures in their framework can be used as an active intrusion detection technique to investigate suspicious network context based on anomaly evolution in Snort.

Generally, Table 2. clarifies some of the recent research for the detection of DDoS attacks using the SDN environment. The KNN al-



gorithm is straightforward to understand and mostly utilized to fix regression and classification issues, but it has the main drawback of being slow when the size of data becomes large. The K-Means algorithm is faster than hierarchical clustering, but it is difficult to predict the K-value. Therefore, the detection trigger mechanism-based rapid response platform has limitations in implementing and operating a rapid response system. The information security defense mechanism (ISDM) is slow compared to the other platforms. It requires a combination of two platforms. With rapid neural network innovation and centralized SDN architecture, we propose a solution using a neural network in an SDN environment.

Proposed Solution

The main goal of the proposed system is to detect DDoS attack flows in the network using a neural network in the SDN environment. This proposed system will be through the following:

- a. Building an SDN environment.
- b. Using Neural Networks
- c. Identifying DDoS attacks.
- d. Measuring performance and comparing the efficiency of the detection rate and accuracy.

This technique for detecting DDoS attacks would protect any organization, business, university, researcher, or government from DDoS attacks by employing or implementing this technique in their network. The detection technique can be used by security analytics, SOC (security operation center) team, and the incident response team. To

form a clear understanding of the functionality of the proposed system. Data flow diagrams (DFD) are used to highlight the transition of data between system processes. The data flow diagram of the detection of DDoS attacks presents the main progress of detection, the flow collection of networks, the feature and characteristic extraction, and classification as shown in Figure 4. As the flow state collection repeatedly sends a request from the flow table to the OpenFlow switch, it also sends all the information from the flow table that came back to it from the switch. The aim of extracting the characteristic principles related to the DDoS attack from the switch flow table and composing the characteristic value is to determine the characteristics of the attack. The classifier analyses characteristic information by using a neural network to identify normal traffic and abnormal traffic. If the traffic is normal, forward the packet, and if attacked by abnormal traffic, drop the packet.

Based on a neural network algorithm, we are developing a system that detects DDoS attacks on the client side using an SDN architecture for either domestic or organizational networks. A loop control system is implemented among three basic architectural components the network, the IDS, and the controller, see Figure 5. A potential DDoS attack may originate from the network, which represents all data traffic. DDoS attacks are detected by the IDS, which analyzes all traffic exchanged across the network. When the IDS detects an ongoing DDoS attack, it notifies the controller. As soon as the controller is notified by the IDS, it sends some new flow rules to the networking devices in the data path to restore the normal operation of the network as quickly as possible. Both the detection and communication phases are crucial to the process.

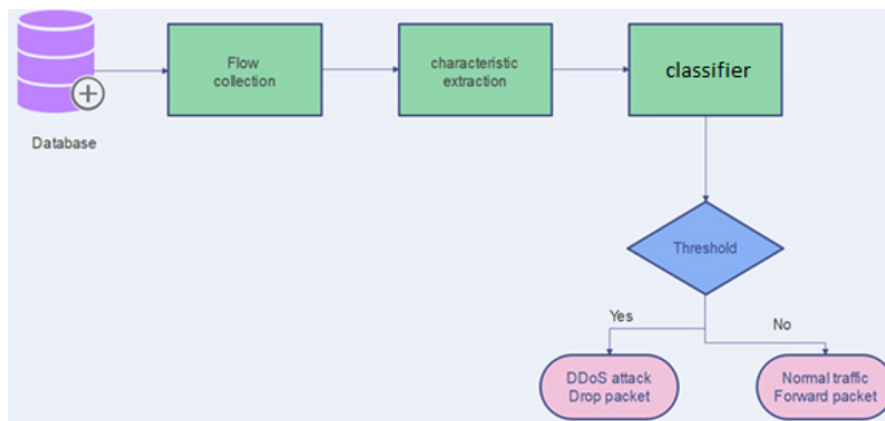


Figure 4: Data Flow Diagram.

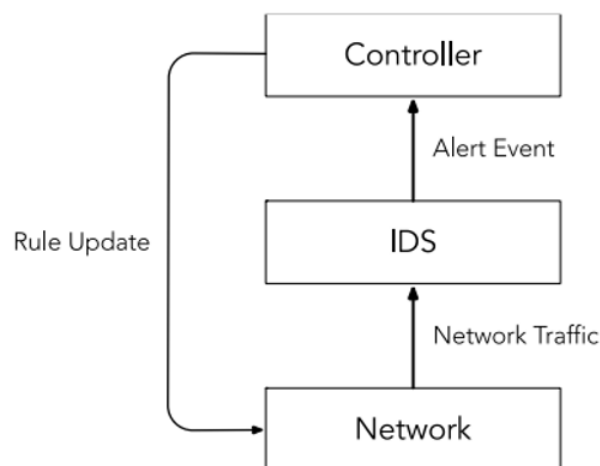


Figure 5: System Architecture of the Proposed Method.

Based on SDN-based neural network analysis, the location stage represents the system's ability to recognize a DDoS attack. The communication stage happens when the IDS cautions the controller about the identified DDoS assault. A controller's relief phase is when a few stream rules are sent to a nearby switch to block the fiendish activities. The stream rules are stored permanently inside the switch.

Figure 6 shows the conceptual work for the current project. We propose combining the usefulness of both an SDN controller and an Interruption Discovery Framework (IDF). The result of combining these two substances is an SDN-Based IDS Screen. Every bundle that arrives at the switch's harbor (i.e., it was gotten within the switch port)

is classified as possessing a place in a stream, which inquiries about the switch. In this way, the switch makes the bundled. In the stream where a bundle has a place, a run-the-show. Initially, the switch has not run the show for that stream, at which point it demands from the SDN controller. As if the packet belongs to a "well-behaved" stream, the SDN controller introduces the right stream. Run the show within the switch, allowing the bundle to continue toward its goal. Additionally, a drop in the show is introduced within the switch. The "bad" stream parcels are disposed of in this final case. To classify streams as "good" or "bad," IDS employs pre-defined rules to classify each ask included in a stream based on the created framework.

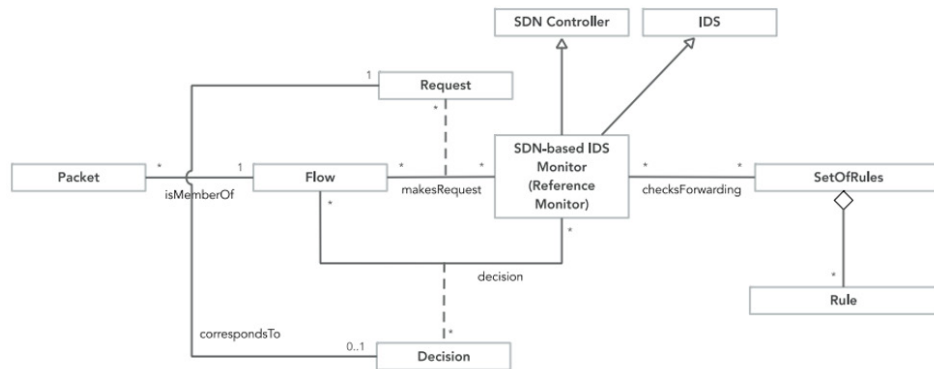


Figure 6: The system's Conceptual Model.

Database Design

The proposed model was trained and tested using the DDoS Evaluation Dataset (CICDDoS2019). Earlier research led to its selection. Various issues have been fixed in NSL KDD, DARPA 99, and CIADA 2007. There are 80 features in the datasets. We are using streaming data on the SDN platform. However, the system has two critical phases: detection and communication. During all stages, data is temporarily processed. This dataset contains network traffic from 12 different DDoS attacks, including NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN Port Scan, and TFTP. A total of 80 different network traffic features are included in the dataset. For this study, we use UDP traffic.

Data Preprocessing

The dataset must be cleaned, standardized, parsed, and reduced before it can be used to train and test the model. All non-numerical values must be converted to numbers. Missing values and outliers are removed. To create a smaller dataset, the datasets are joined and transformed using Principal Component Analysis (PCA) while maintaining most of the information. PCA is a technique used to reduce the dimensionality of large datasets. Most of the information from the large set is retained by converting a large set of variables into a smaller set.

Did you perform data processing and use PCA ?? no

Experimental Setup

For implementation purposes, Ubuntu virtual machine employs VirtualBox, and to create a network for the detection attack, we have used Mininet. The controller that we use is the RYU controller, and Wireshark is used for capturing packets. With Mininet, you can simulate a large network on a single computer. By running actual kernel, switch, and application codes, Mininet allows users to quickly create an actual virtual network on a personal computer by running actual kernel, switch, and application codes. DDoS attacks have also been

performed on Mininet. Ryu Controller is an open, software-defined networking (SDN) controller designed to increase network agility. Mininet uses the Ryu controller, which is a modular, SDN-based framework for creating network applications. Through it, network operators can interact directly with packet headers at multiple levels of the OSI model, including L2, L3, and L4 protocols. To update the OpenFlow switch, the user chooses match fields and operations from the provided options. In our SDN environment, we used the RYU controller.

Wireshark is used to capture and analyze TCP in this experiment. TCP flooding (DDoS) attacks send packets to the victim server. It was used to analyze benign and DDoS traffic. Here we are explaining packet behavior.

a. Wireshark: The Wireshark Network packet analyzer shows network traffic in real-time and is used by ethical hackers as a network security tool. In our experiment, we show the captured and analyzed TCP using Wireshark. The packet behavior of TCP flooding (DDoS) attacks is that the packets are sent to the victim server. We used it to analyze benign and DDoS traffic.

b. Visual Studio is a source code editor, a computer program for developers. We used it to run SOM and XGBoost algorithms.

c. Xterm: xterm is the standard terminal emulator of the X Window System, providing a command-line interface within a window. Within the same display, multiple xterm instances can run concurrently, each providing input and output for a shell or another process. During the experiment, we used command-line interfaces between hosts to perform the attack from host2 to host4.

d. Google Colab: Collab for short, is a web-based application by Google for executing online code on its other service, Drive (Google Drive). It allows you to combine executable code and rich text into a single document known as a Notebook Document, along with images, HTML, LaTeX, and more. A notebook document is made up of cells just like Jupiter Notebooks and can contain code (mainly Python),

text with the powerful Markdown language capability, images, visualizations, and more. Collab is backed by Google servers and runtimes where the code is executed without the need for any kind of additional setup on your device. It also comes packed with a complete library of searchable code and an interactive and illustrated statistical visualization library called Altair for Python. We used to write, run, and test our code and results. The programming language that we use is Python, which contains an open-source library and tools to develop models for ML. And the packages that we have used for the algorithm script are listed below in Table 3.

To show how Mininet can be extended, an experimental tool was created in Mini Edit. We have used Mini Edit to create a custom network

topology as shown in Figure 7. After creating the custom otology, we need to configure the controllers by setting a unique port number for each controller. There are three controllers. We used the default Open-Flow Reference controller that comes with Mininet in this example. Nevertheless, each controller needs to be configured so that it uses a different port. Choose Properties from the menu that appears when you right-click each controller. By default, each controller has port 6633. As a result, controllers c0, c1, and c2 use ports 6633, 6634, and 6635 respectively Use the remote control whenever possible. Then, we customize Mini Edit's preferences. MINI Edit preferences are stored in the topology file associated with each saved scenario, so different preferences may apply to different saved scenarios. Finally, we need to set up the Ryu controller. Installing Ryu is quite easy.

Table 2: Python libraries.

Libraries	Used for
NumPy	It is useful for performing mathematical and logical operations on arrays
Pandas	It is useful for analysis purposes in ML software. It can import data files in different formats.
Pip	It is a tool for installing additional libraries in Python.
Sklearn	(Scikit-learn) In Python, this library is the most useful for machine learning. We used it to separate the dataset for training and testing.
Hashlib	The Python hashing function is used here to store the hash of the password in the DB
Keras	Keras is used for creating deep models which can be produced on smartphones. Keras is also used for distributed training of deep learning models. Keras is used by companies such as Netflix, Yelp, Uber, etc.
Plotly	Plotly is an open-source library that provides a list of chart types as well as tools with callbacks to make a dashboard. The charts which I have embedded here are all made in the chart studio of plotly. It helps to embed charts easily anywhere you want.

Table 3: Comparison of our model with existing algorithms.

Techniques	Platform	Accuracy (%)
NB (Naive Baye Algorithm) (Ajith,2021)	CPU	57
Booster (Ajith,2021)	SDN/ IoT networks	84
RF (Random Forest Algorithm) (Ajith,2021)	SDN	86
SVM (Support Vector Machine) (Ajith,2021)	SDN/ CPU contains one of the available	93
LR algorithm (Ajith,2021)	SDN	95
Our Algorithm that was used by Keras	SDN	98

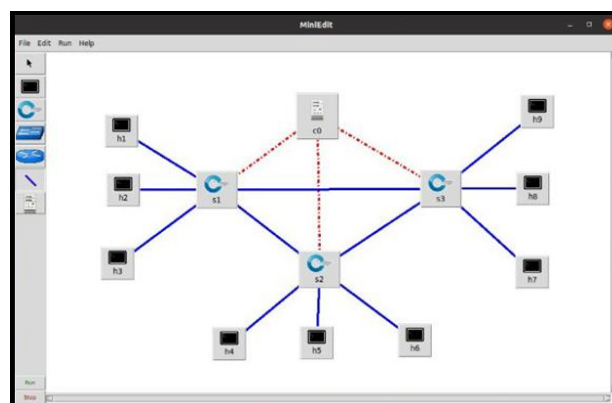


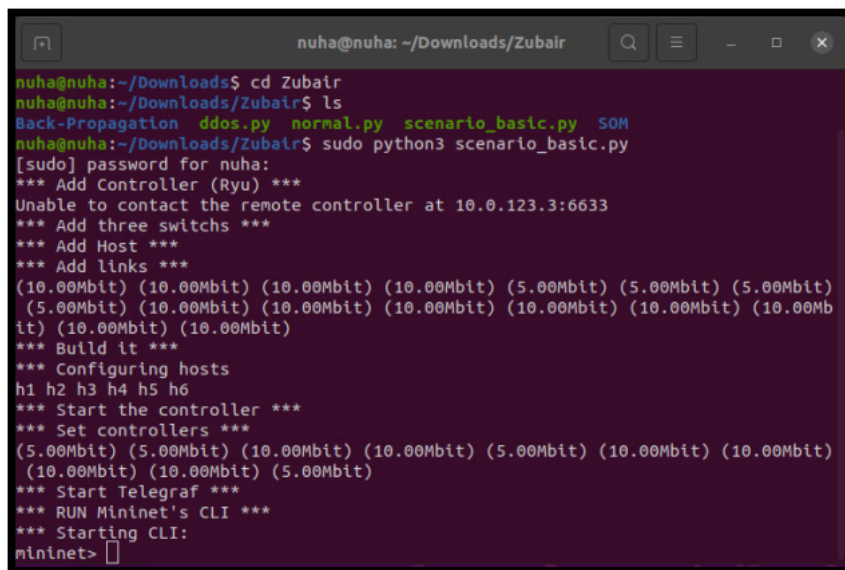
Figure 7: Network Topology.

To perform the DDOS attack, we ran the controller as a command executed in the terminal, sing file named scenario_basic.py, as shown in Figure 8. Then, Execute the scenario file to build the network that we exported by Mini edit. To attack Host4 using Host1, we just create the h1, h2, h4 hand Ost machines using termtime to launch hping3 from Host1 as shown in Figure 10.

Then, If we now, try to ping Host4 from Host2 we'll fail horribly as shown in Figure 11, which means that Host4 is under DDoS attack. Before running the Neural Network algorithm to detect DDoS attacks, we used SOM and XGBoost algorithms and we processed them both separately to see their accuracy in detecting the attack stack. SOM algorithm is proceeding using the Kaggle website and the accuracy was very low 57.14%, see Figure 12. The accuracy result while processing the XGBoost algorithm was 88%, see Figure 13. After processing both algorithms, we found that the results were not as we expected for the SOM algorithm and XGBoost. In a common-sized network with many switches for the external network, observing all traffic for these switches and detecting DDoS attacks requires each gateway to send traffic to the controller to detect and reorganize SOM due to a single point of aggregation. On the other hand, our experiment uses a boosting technique called XGBoost. It appears that the accuracy rate is low. In addition, we decided to implement a neural network model for detecting DDoS attacks. don't understand this. I have adjusted to be more clear

We write the code for the detection of DDoS by using neural networks in python, a file named Sarah Nuha_algo.py. (we need to upload the code in setup and provide the link) I can't past all the code here it's too long I will attach a link or code file. The dataset that we have used for training and testing the model can be found at. The CIC-DDoS DDos Evaluation Dataset was used as the basis for our analysis. CIC has created a new one to correct all deficiencies in the existing dataset. They developed a new method for detecting and classifying families based on the dataset generated. We conclude by providing the key features and weights required to detect various types of DDos attacks.

CICDoS contains benign and the most modern DDos attacks, representing actual real-world data (PCAPs). The dataset consists of 3136802 rows and 80 columns. The dataset also contains the results of the network traffic analysis using CICFlowMeter-V3, labeled by timestamps, IP addresses, destination IP addresses, ports, protocols, and attacks. Several different reflection DDos attacks are included in this dataset, including Port Map, NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, SNMP, NTP, DNS, and SNMP. Several attacks were also conducted during this period. A total of 12 DDos attacks are included in this dataset, including DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, and TFTP. We used UPD in our experiment. However, because we could not test all types of DDos attack datasets due to changes, we encountered multiple errors after running all types of DDos attack datasets. Due to the machines' inability, we ran the code on the Google collab server instead.

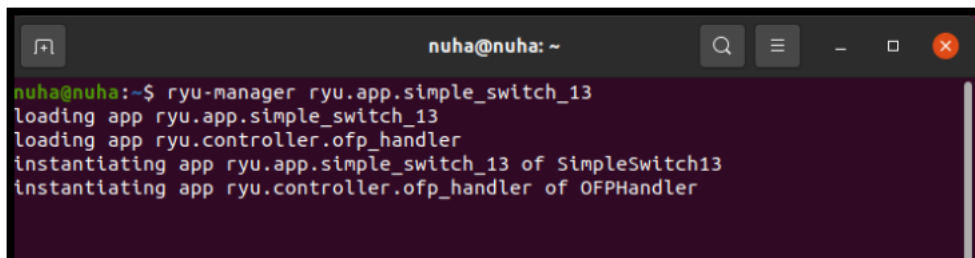


```

nuha@nuha: ~/Downloads/Zubair
nuha@nuha:~/Downloads$ cd Zubair
nuha@nuha:~/Downloads/Zubair$ ls
Back-Propagation ddos.py normal.py scenario_basic.py SOM
nuha@nuha:~/Downloads/Zubair$ sudo python3 scenario_basic.py
[sudo] password for nuha:
*** Add Controller (Ryu) ***
Unable to contact the remote controller at 10.0.123.3:6633
*** Add three switches ***
*** Add Host ***
*** Add links ***
(10.00Mbit) (10.00Mbit) (10.00Mbit) (10.00Mbit) (5.00Mbit) (5.00Mbit) (5.00Mbit)
(5.00Mbit) (10.00Mbit) (10.00Mbit) (10.00Mbit) (10.00Mbit) (10.00Mbit) (10.00Mbit)
it) (10.00Mbit) (10.00Mbit)
*** Build it ***
*** Configuring hosts
h1 h2 h3 h4 h5 h6
*** Start the controller ***
*** Set controllers ***
(5.00Mbit) (5.00Mbit) (10.00Mbit) (10.00Mbit) (5.00Mbit) (10.00Mbit) (10.00Mbit)
(10.00Mbit) (10.00Mbit) (5.00Mbit)
*** Start Telegraf ***
*** RUN Mininet's CLI ***
*** Starting CLI:
mininet>

```

Figure 8: Run the Controller using the Scenario basic python file.



```

nuha@nuha: ~
nuha@nuha:~$ ryu-manager ryu.app.simple_switch_13
loading app ryu.app.simple_switch_13
loading app ryu.controller.ofp_handler
instantiating app ryu.app.simple_switch_13 of SimpleSwitch13
instantiating app ryu.controller.ofp_handler of OFPHandler

```

Figure 9: Loading app RYU_smiple_switch_13.

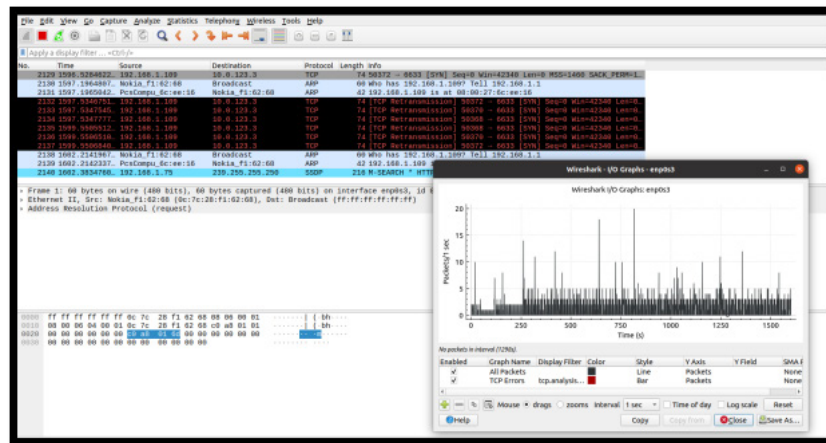


Figure 10: Analyzing benign and harmful traffic by Wireshark.



Figure 11: host1 ping host4.



Figure 12: Host2 is pinging host4, which is under DDoS attack.

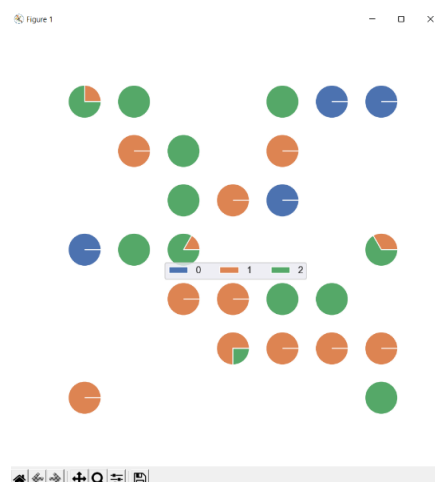


Figure 13: Graphical Accuracy Rate for SOM Algorithm.

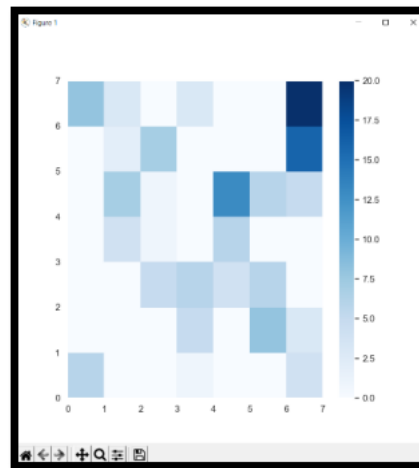


Figure 14: Graphical Accuracy Rate for XGBoost Algorithm.

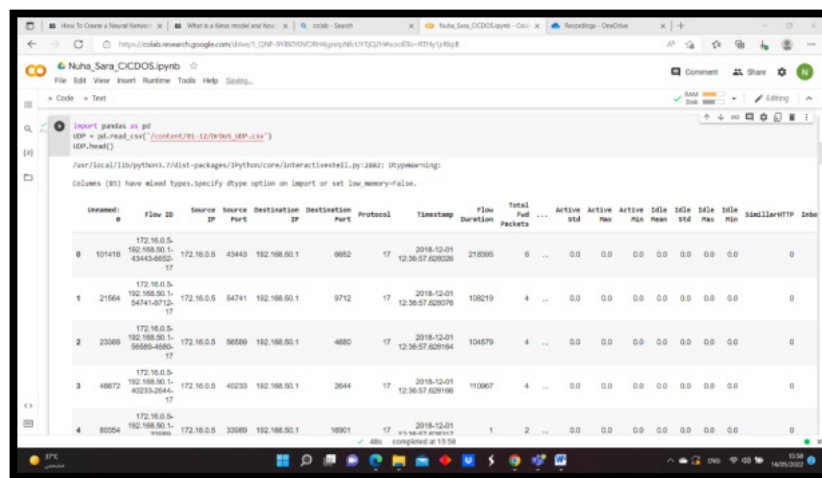


Figure 15: Pandas dataframe

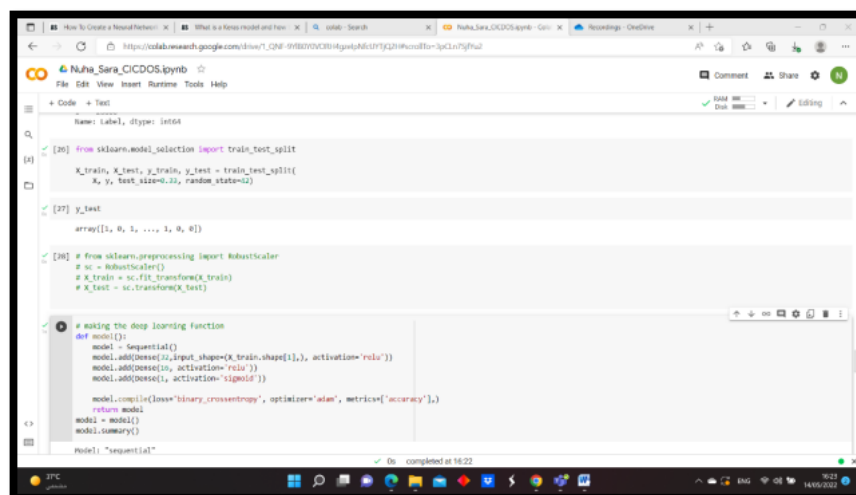


Figure 16: Using Keras library to create our model.

Experimental Results

We download the dataset in Google Collab Interface, then we Import the necessary libraries and import the pandas data frame using the DrDos UDP csv file from the dataset, as shown in Figure 14, and save it into the UDP variable. Now, we can do further preprocessing using the UDP variable.

Then, we extracted the column names from the dataset and renamed the columns whose names are not valid for our task. We remove some columns that are either unnecessary or detrimental to our training. The dropped columns are: 'Unnamed: 0', 'Flow ID', 'Source IP', 'Destination IP', 'Timestamp', 'Similarity', 'Flow Bytes/s', and 'Flow Packets/s'. Next, we check the number of null values. There are no null

values in the dataset. Then, we encoded our target column. We indicate our classes: "benign" as 0 and "DrDoS_UDP" as 1. This encoding will be needed for the training.

Dealing With Class Imbalance Problem

We notice that there are very few samples for the Benign class compared to the DrDoS_UDP class. So, we must deal with this imbalance; otherwise, our training model will be biased. So, we use the Sklearn resample function to resample the two classes. Sklearn is the most useful and robust library for machine learning in Python, it provides a selection of efficient tools for machine learning and statistical modeling, including classification, regression, clustering, and dimensionality reduction.

After that, two classes have the same number of rows. We choose all the columns except the dropped column and the target column as our X data. Only target data is assigned to the Y variable. Then, we split the whole dataset using the Sklearn train test split function. A total of 67% of the dataset is selected for training and the other 33% is set for testing.

Designing the Neural Network Model from Scratch

There are two ways to create a neural network in Python:

From Scratch, this can be a good learning exercise as it will teach you how neural networks work from the ground up. Using Neural Network Library: packages like Keras and TensorFlow simplify the building of neural networks by abstracting away the low-level code. If you are already familiar with how neural networks work, this is the fastest and easiest way to create one.

In Keras, we can define TensorFlow neural networks by specifying the attributes, functions, and layers of your models. There are several APIs that you can use with Keras to develop neural networks. That we used sequentially. An API that lets you build models layer by layer for most problems. The interface is relatively straightforward (just a list of layers), but it is limited to layers with single inputs and outputs. The model is developed by creating instances of the sequential class using the Sequential API. Input variables are one and two, the hidden layers have two neurons, and the output layer has one binary output.

Then, we used the Keras library to design the algorithm. From scratch, we designed a sequential neural network algorithm. Input training data is fed to the first layer. We set 32 as the first unit value. Units are one of the most fundamental and required parameters of the Keras dense layer, defining the size of the dense layer's output. It must be a positive integer, as it reflects the number of dimensions of the output vector. In the second layer, the unit size is set to 16. For both layers, we used the relu activation function. At the last layer, the unit layer is set to 1, because we need to perform binary classification here and the sigmoid activation function is used. We use Adam optimizers to select the binary cross-entropy loss function.

Then, our neural network algorithm is trained with 100 epochs where the batch size is set to 100. The model learns very well, and we achieved 99.9% training accuracy, where validation accuracy was around 99.88%. We use the fit method from the Keras library, which is used for creating deep models which can be produced on smartphones. Keras is also used for distributed training of deep learning models.

We test the model algorithm's performance with the test set. We achieved 99.74% testing accuracy. That indicates our model performs very well. We used a ReLU activation function for every layer to achieve better results. The model achieved an overall accuracy of 99.74% when trained for 100 epochs. Furthermore, we changed the number of hidden layers, iteration, channels per layer, and the activation function for each. With two hidden layers, we got the best results. When the number of hidden layers is increased, the model accuracy

remains the same, but the training time is longer. As a result, we propose using two layers. Therefore, the three layers are more convincing to provide reasonable results. Comparing the performance of proposed models with other classical ML algorithms. In comparison to other benchmarking algorithms, our proposed approach performs the best. Comparison of our model with existing algorithms:

Conclusion

We propose in this paper a completely systematic detection approach for the DDoS attack. First, we selected the CIC-DDoS dataset from the sdn datasets repository that contains information about the DDoS attacks. Then, Python and Google Collab notebook. After data normalization, we applied the proposed, supervised, machine learning approach. The classification algorithms Neural Network and XGBoost were then used. And after testing, we test our model algorithm performance with the test set. We achieved 99.74% testing accuracy, which has enabled us to check, measure, and compare the effectiveness of the detection rate and accuracy of the proposed solution with the previously available results. The project can be further developed by implementing it as a browser extension or system application. Researchers can also work on improving the accuracy rate of the ML models by trying diverse sets of features. Test dataset that includes all the types of DDoS attacks. Moreover, we will investigate how no supervised learning algorithms will affect the detection of DDoS attacks when non-labeled datasets are considered.

References

1. Tonkal Ö, Polat H, Başaran E, Cömert Z, Kocaoglu R (2021) Machine learning approach equipped with neighborhood component analysis for DDoS attack detection in software-defined networking. *Electronics* 10(11).
2. Zargar ST, Joshi J, Tipper D (2013) A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. *IEEE Communications Surveys and Tutorials* 15(4): 2046-2069.
3. Hu D, Hong P, Chen Y (2017) FADM: DDoS Flooding Attack Detection and Mitigation System in Software-Defined Networking. 2017 IEEE Global Communications Conference GLOBECOM 2017 - Proceedings 2018 :1-7.
4. Mahjabin T, Xiao Y, Sun G, Jiang W (2017b) A survey of distributed denial-of-service attack, prevention, and mitigation techniques. *International Journal of Distributed Sensor Networks* 13(12).
5. Yang X, Han B, Sun Z, Huang J (2017) SDN-Based DDoS Attack Detection with Cross-Plane Collaboration and Lightweight Flow Monitoring. 2017 IEEE Global Communications Conference, GLOBECOM 2017 -Proceedings 2018: 1-6.
6. Sangodoyin A, Modu B, Awan I, Pagna Disso J (2018) An Approach to Detecting Distributed Denial of Service Attacks in Software Defined Networks. *Proceedings - 2018 IEEE 6th International Conference on Future Internet of Things and Cloud FiCloud* 436-443.
7. Sidhu N, Saluja KK, Sachdeva M (2012) DDoS Attack's Simulation using Legitimate and Attack Real Data Sets.
8. Lin HC, Wang P Implementation of an SDN-based Security Defense Mechanism Against DDoS Attacks. In Kunda Rd YongKang Dist (Issue 195).
9. Polat H, Polat O, Cetin A (2020) Detecting DDoS attacks in software-defined networks through feature selection methods and machine learning models. *Sustainability* 12(3).
10. Tuan NN, Hung PH, Nghia ND, van Tho, N van Phan et.al. (2020a) A DDoS attack mitigation scheme in ISP networks using machine learning based on SDN. *Electronics* 9(3).
11. Wang W, Ke X, Wang L (2018) An HMM-R approach to detect L-DDoS attack adaptively on SDN controller. *Future Internet* 10(9)
12. Yu Y, Guo L, Liu Y, Zheng J, Zong Y (2018) An efficient SDN-Based DDoS attack detection and rapid response platform in vehicular networks. *IEEE Access* 6: 44570-44579.



13. Tuan NN, Hung PH, Nghia ND, van Tho, N van Phan et.al. (2020b) A DDoS attack mitigation scheme in ISP networks using machine learning based on SDN. *Electronics* 9(3).
14. Tan L, Pan Y, Wu J, Zhou J, Jiang H et.al. (2020) A New Framework for DDoS Attack Detection and Defense in SDN Environment. *IEEE Access* 8: 161908-161919.
15. Elsherif AA, Aldaej AA (2020) DDOS Botnets Attacks Detection in Anomaly Traffic: A Comparative Study. *Journal of Information Security and Cybercrimes Research* 3(1): 64-74.
16. Kato K, Klyuev V (2014) An Intelligent DDoS Attack Detection System Using Packet Analysis and Support Vector Machine.
17. Mahjabin T, Xiao Y, Sun G, Jiang W (2017a) A survey of a distributed denial-of-service attack, prevention, and mitigation techniques. *International Journal of Distributed Sensor Networks* 13(12).

