TALLINN UNIVERSITY OF TECHNOLOGY
MASTER THESIS
05/2022

Deep Learning-Based Detection of DDoS Attacks in Software-defined Networks

SEYED MOHAMMAD HADI MIRSADEGHI
195463IVCM


TAL
TECH
PRESS

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technology
Department of Software Science

The dissertation was accepted for the defence of the degree of Master of Science in May 2022

Supervisor: Professor Hayretdin Bahsi,
Department of Software Science School of Information Technology,
Tallinn University of Technology
Tallinn, Estonia

Co-supervisor: Professor Wissem Inoubli,
Department of Software Science School of Information Technology,
Tallinn University of Technology
Tallinn, Estonia

Defence of the thesis: May-June 2022, Tallinn

Declaration:
Hereby I declare that this master's thesis, my original investigation and achievement, submitted for the Master of Science degree at Tallinn University of Technology, has not been submitted for any academic degree elsewhere.

Seyed Mohammad Hadi Mirsadeghi
195463IVCM

signature

TALLINNA TEHNIKAÜLIKOOL
MEISTERÖÖ
05/2022

sügav õppimispõhine ddos-rünnakute tuvastamine tarkvaraga määratletud võrkudes

SEYED MOHAMMAD HADI MIRSADEGHI
195463IVCM

Abstract

Distributed Denial-of-Service attacks' harm and traffic are at an all time high. Even though it was more than 20 years ago when we observed the first instance of this broad attack class, recent studies show that DDoS attacks have become more advanced and sophisticated with time. There are reports of attack traffic in the Terabites and use of Amplification techniques highlighting the utmost imporance of proper DDoS detection models. Deep Learning and Signal Processing offer high precision modeling capability with Artificial Neural Networks that learn the latent space representaion and structure of network flow instances and make for successful Network Intrusion Detection models. Moreover, Software-Defined Networking has revolutionized network management and network programmiblity and promises the next generation Internet (5G). Hallmarks of SDN such as the centralized architecture and network-wide visibility offer advantages in Intrusion Detection and alter the Cyber Threat Landscape at the same time. In this study, We Propose two Deep Learning models for detection of DDoS attacks in the InSDN dataset that achieve remakrable scores of accuracy and precision in the 99th percentile. Furthermore, we ask the question: "Is it possible to detect more DDoS attacks when IDS models cooperate?".To answer this question, We demonstrate that the number of false negatives in our DDoS detection scheme drops drastically when models cooperate.

Acknowledgements

This project would not have been possible without the support of many people. Many thanks to my supervisor Professor Dr. Hayretdin Bahsi who offered guidance, support and encouragement all along. Also thanks to Dr. Wissem Inoubli who provided me with valuable feedback. This work is dedicated to my Father who shares my Soul and Passion for Science.

Abbreviations

SDN	Software defined Networking
IDS	Intrusion Detection System
DDoS	Distributed Denial-of-Service
NIDS	Network Intrusion Detection System
IXP	Internet Exchange Point
SPOF	Single Point of Failure
SD-WAN	Software-Defined Wide Area Network
QoS	Quality of Service
IoT	Internet of Things
BFA	Brute-Force Attack
LSTM	Long Short-Term Memory

Contents

1 Introduction

Future Internet promises better security and proposed models for future networks include elements of network programmibility and a centrally managed control plane at their core. Software-Defined Networking has revolutionized network management through the physical seperation of the control plane from forwarding devices and promises the next generation Internet (5G). Through the decoupling of the control and data planes in the SDN paradigm, all network intelligence and control logic is migrated from the network devices to a logically centralized software-based entity known as the network controller. The network controller resides in the control plane where centralized control and network management functions instruct forwarding behavior to all the elements distributed in the infrastructure. The centralized characteristic of SDN implies that the network controller is always aware of the network state and that all traffic flows are passed to the controller at least once in the network lifetime for the definition of forwarding behavior [12].

SDN nurtures the conception of network programmability, therefore network security functions such as Intrusion Detection Systems are embedded as software applications that can be either installed on top of the controller or deployed as independent data consumer functions. The centralized SDN architecture and proactive packet processing alter IDS research in SDN in the following ways, to name a few:

- the network controller and the IDS in turn, have full visibility into the network state both in breadth and depth.
- The IDS, typically implemented at the network barrier in traditional networks, is implemented as an application on top of the network controller which changes IDS's vantage point with respect to potential attack traffic.
- while highly suitable for security applications, SDN technology poses many vulnerabilities and threats that are challenging to address [7].

Cyberattacks have become more frequent and devastating [2] [5] [20]. Among the most popular cyberattacks are those that target online service. The first Distributed Denial-of-Service attack was observed for the first time more than twenty years ago and yet recent studies show that these attacks have become more advanced and sophisticated with time. the response from the industry was to introduce DDoS detection and mitigation platforms deployed at various locations in the internet [13, 14] . With the advent of SDN, efficient DDoS detection solutions with low numbers of false negatives are ever more important in the cyber threat landscape in future Internet.

Network Intrusion Detection Systems (NIDS) use network traffic data to find anomalies (abnormal activity). Whether they define intrusion(anomaly) as deviation from normal behavior or network traffic data is inspected against already-developed signatures to spot anomalous behavior (a.k.a. misuse-based intrusion detection), developing a data-driven approach is mandatory in solving this problem. Machine Learning methods, Deep Neural Networks in particular, have proven to be an essential tool when processing loads of network traffic data.

In order to detect intrusions over the network, We usually examine flows of traffic rather than single packets. Even more so in the case of DDoS attacks because they occur over consecutive periods of time with high traffic volume and increasingly high numbers of packets per second. Typical DDoS attacks can generate large traffic volumes and involve hundreds of reflectors. Assuming that it is unlikely for an Internet client to recieve traffic from many sources with the same port number at a high traffic rate, [16] proposes a filter method that has proven helpful in differantiating between attack and benign traffic

from the vantage point of Internet Exchange Points (IXP). To characterize DDoS attacks in network traffic, we consider coarse-grained properties like Flow Duration, Packet count, and Packet rate as well as packet details such as distribution of port numbers, protocol types, and packet size[18]. Moreover, we take advantage of spatial features and temporal correlations of network flows that help identify DDoS from benign traffic.

In this thesis project, we analyze attack traffic vs normal traffic in Software-defined networks to characterize DDoS attacks, propose efficient deep learning models for DDoS detection and ultimately assess the potential benefits of cooperative DDoS detection in SDN.

1.1 Motivation

SDN features like network-wide visibility, centralized network intelligence and network programmability reshaped the way packet forwarding and basic network control duties are performed in programmable networks. However, these features and the SDN architecture itself introduce new security risks and attack vectors that are not present in conventional network deployments. Therefore, we can see that SDN security has a twofold connotation: in the first place we can improve network security through SDN features to protect, react and provide mitigation schemes against well-known security risks and in second place, we ought to design a more secure SDN architecture that addresses the new attack vectors [17]. The new attack surface introduced by SDN is due to the inherent alterations to network components and the relationships between them. For example, the Centralized architecture brings about a single point of failure (SPOF). In other words, if the network controller is compromised by an adversary, the entire network may be in jeopardy. Moreover, SDN elements themselves (like a DNS application) may be used as reflectors in Amplification DDoS attacks.

Here, we go over important SDN features that can be leveraged to implement a variety of security applications across the network.

- `Dynamic Flow Control` can benefit security by enforcing security middleboxes as composition of different sets of flow control rules that are instructed throughout the network [17]. Moreover, SDN dynamic flow control can be leveraged to install security applications such as firewalls, access control lists, and Intrusion Detection Systems on top of the network controller or bound to the controller through the northbound interface.

- `Network-wide Visibility` means the network controller is aware of the state of any network element deployed anywhere at any time. Network-wide Visibility coupled with centralized flow control makes attack detection and prevention all the more straight-forward in SDN.

- `Network programmability` supports the process of harvesting intelligence from existing Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS). This approach, followed by analysis and centralized reprogramming of the network can render SDN more robust to malicious attack than traditional networks [24].

Security in Software=Defined Networks is paramount. B4 [5], [6] is Google's globally deployed software-defined inter-datacenter wide area network. While datasets used by most current research literature are variants of the KDD dataset [7], Network traffic data from B4 can be of assistance in understanding security within SDN. Network traffic data is perhaps the most salient source to monitor. This data can be used to effectively design and train Machine Learning models in intrusion detection applications.

In Recent years, Machine Learning methods, Deep Neural Networks in particular have proven to be useful in solving networking problems. For Example, Pensieve [8] uses reinforcement learning techniques to generate adaptive bitrate connection latency by sending data directly when establishing algorithms.

The Capabilities of SDN such as software-based traffic analysis and global view of the network along with dynamic updating of forwarding rules facilitate detection of attacks in SDN [27].

By some accounts, Google's network delivers more than 25% of internet traffic. Googe infrastructure is critical and its security is one of the top priorities. The motivation behind this research project is to address security, intrusion detection in particular, within SD-WANs. SD-WANs such as Google's Espresso and B4 are in most cases important backbone networks that the organization's infrastructure relies on. In such a context, the virtue of reliability, QoS, and security are paramount.

1.2 Research Objective and Questions

The aim of this research project to address questions and challenges on the subject of network intrusion detection within SDNs. one of the most effective approaches in Network Intrusion Detection is to adopt a data-driven approach. For our study, we leverage InSDN [7]: a recent SDN attack-specific dataset. The InSDN dataset contains a total number of 343,939 network flow traces where normal traffic brings 68424 instances and attack traffic is distributed across 5 different attack classes with DDoS and Probe attacks contributing 73529 and 61757 instances, respectively [7]. The remaining one percent of attack traffic is made up of 1145 DoS attack instances, 295 instances of BFA attacks and only 17 instances of U2R.

Our goal is to analyze the available data to characterize DDoS attacks, propose efficient models for DDoS detection and ultimately assess the potential benefits of cooperative DDoS detection in Software-defined networks.

Our research questions are:

- Is it possible to detect more DDoS attacks when IDS models in SDN cooperate?
- What vulnerabilites can be spotted through static analysis of the SDN security?
- Can Deep Neural Networks be effective in detecting intrusion over the software-defined networks?

1.3 Scope and Goal

History of SDN attacks is uknown. this is mainly due to the fact that it is an evolving technology.the InSDN dataset is the most recent publicly available dataset to the Intrusion detection problem. data collected from globally deployed SDNs such as Google B4 and cloud computing infrastructres can reveal more about the signature characterstics of intrusion detection is software-defined Networks.

Network Intrusion Detection Systems (NIDS) spot attack traffic by finding anomalies in network traffic data.Whether it defines intrusion as deviation from normal behavior or it's misuse-based where the network state is checked against already-developed signatures.Network Intrusion Detection Systems are one of the industry's main response to the threat posed by high-impact cybersecurity attacks such as DDoS attacks.

Machine-learning is one of the most promising network intrusion detection techniques. Moreover, Deep Learning techniques have shown remarkable results.The goal of this study

is to propose deep learning models for the problem of DDoS detection in Software-defined Networks and answer the question: "Is it possible to detect more DDoS attacks in SDN when models cooperate ?"

1.4 Novelty

Deep Learning Models have been deployed for Intrusion Detection to understand the spatial and temporal features of network traffic data. Even though Software-defined Networking is highly deployed in datacenters, it is still an evolving technology the research community can benefit from more intrinsic data. These Deep Learning-based Intrusion Detection models are composed of multiple layers of abstractions and deep neural connections. Some studies use sequence modeling formulation and include elements of Long Short-term Memory units that learn the temporal stucture of time-series network statistics. While Results from simillar studies indicate an overall adequate performance. In addition to the numerically superior performance metrics of our proposed models as compared to similiar studies, we were able to significantly reduce the number of false negatives through cooperation.

2 Related Works

In the past, Statistical Approaches have been used for DDoS Attack Detection. This paper [8] presents methods to identify DDoS attacks by computing entropy and frequency-sorted distributions of selected packet attributes. The DDoS attacks show anomalies in the characteristics of the selected packet attributes.

Machine learning algorithms are used to solve complex problems in many fields [112]. These algorithms are also applied for detection of DDoS attacks, and it has been found that they are better than signature-based detection techniques[113].

Deep learning allows computational models composed of multiple layers to learn the data structure with multiple levels of abstraction and improve the intelligence of intrusion detection systems (IDS).

The history of software-defined networks is mostly uknown. InSDN [7] is the first-of-its kind dataset on the subject of intrusin detection in SDN. while contributory to the research community, it is a synthetic dataset generated by using an SDN testbed and self-initited attacks.

The InSDN dataset is the most recent state-of-the-art Intrusion Detection dataset in Software-Defined Networks. Authors of [26] propose a deep learning model based on an Autoencoder and a Multi-Layer Perceptron to classify different kinds of intrusions in SDN. Moreover, A recent study [1] employs elements of Long Short-term Memory Units as part of their classification model to better learn the temporal features of the dataset.

Recent studies have demonstrated the high accuracy of deep learning methods in detecting IoT botnets. Authors of this paper [19] show that it is possible to induce high accurate unsupervised learning models with reduced feature set sizes, enabling to decrease the required computational resources. Training one common model for all IoT devices, instead of dedicated model for each device which is another design option that is evaluated for resource optimization. The following paper [10] applies hybrid feature selection method for Machine Learning-Based Botnet Detection in IoT Networks.

Moreover, Authors of this Paper [9] propose a distributed approach to defending against distributed denial of service attacks by coordinating across the Internet. In this approach, DDoS defence systems are deployed independently in the network to detect DDoS attacks where information is exchanged between these nodes via a gossip based communication mechanism.

3 Dataset

Even though SDN is increasingly deployed in datacenters, cloud computing infrastructure, and globally deployed WANs, the history of SDN attacks in still unknown [7]. While classical intrusion detection datasets such as KDD'99 [6], NSL-KDD [25] provide us with cases of DDoS attacks vs normal traffic, they do not reflect the signature characteristics of SDN such as the Centralized architecture and full visibility into the network state to name a few. Actual Traffic data captured from datacenters and globally deployed WANs such as Google B4 [11] and Espresso [28] can shed a light on the specifics of DDoS detection in SDN. To the best of our knowledge however, there are still no publicly available datasets of real DDoS attacks in Software-defined Networks.The Booter [3] dataset provides the research community with real instances of DNS Amplification DDoS attacks. Even though helpful in understanding how to detect DDoS attacks in traditional networks, using a non-specific SDN datast may cause compatibility problems as the attack vectors would not resonate with the SDN architecture.

For our study, we leverage InSDN [7]: a recent SDN attack-specific dataset. The InSDN dataset contains a total number of 343,939 network flow traces where normal traffic brings 68424 instances and attack traffic is distributed across 5 different attack classes with DDoS and Probe attacks contributing 73529 and 61757 instances, respectively [7]. The remaining one percent of attack traffic is made up of 1145 DoS attack instances, 295 instances of BFA attacks and only 17 instances of U2R. This dataset provides flow-level features such as flow duration, inter-arrival time, number of packets, and number of bytes.

Our goal is to analyze the available data to characterize DDoS attacks, propose effective deep learning models for Intrusion detection (Detection of DDoS attacks as well as instances of DoS, U2R, Probe, and Brute-Force attacks) and ultimately assess the potential benefits of cooperative DDoS detection in Software-defined networks.

4 Problem Formulation

4.1 Network Architecture

Authors of InSDN [7] data set represent the network topology by creating four virtual machines (VMs) using VMware Workstation on Windows 10.

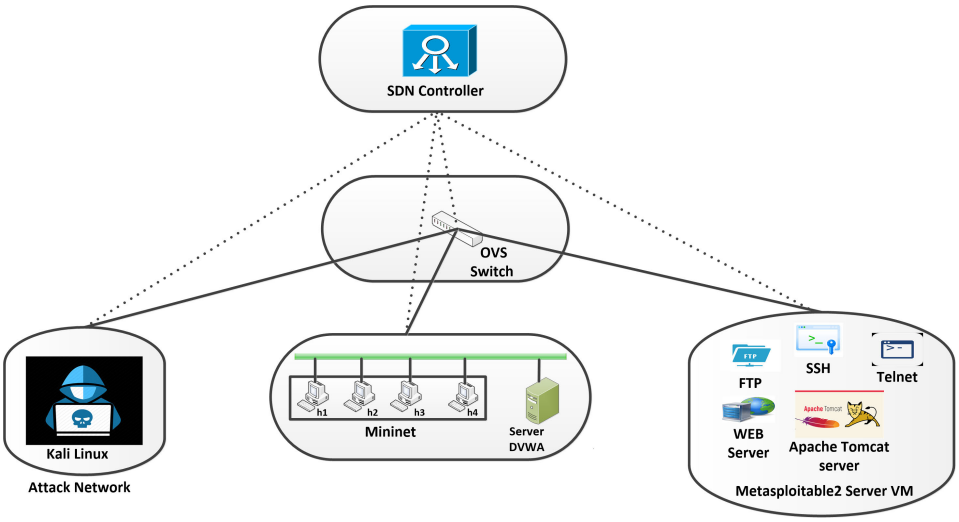

Figure 1 – InSDN [7] : Virtual SDN testbed network architecture

4.2 Data Generation Methodology

The centralized view of the SDN network and separation of the data plane from the control plane creates a new opportunity for the attacker to carry out various types of attacks compared to the conventional network [7]. The attacker may launch an attack against the SDN controller or even the communication links between the SDN controller and OpenFlow switches.

The DDoS attack scenarios in the InSDN dataset are TCP-SYN Flood, UDP Flood, and ICMP Flood attacks.

The remote exploitation and backdoor attacks represent the U2R scenario in the InSDN dataset. These malicious activities can bring about serious damage to the network.

4.3 Limitations and Considerations

The InSDN dataset, while providing the first novel SDN intrusion detection dataset, only includes synthetic attack traffic generated in the SDN testbed characterized by its authors. We understand that the quality of a good model depends on the quality of the dataset and how well it represents the phenomenon under study.

Many catastrophic real-world cases of DDoS attacks incorporate amplification attacks and reflector elements across the network. for example, the attacker may take advantage of the centralized SDN architecture and target a network service as reflector within a DDoS amplification attack scheme.

The InSDN testbed was implemented using only ONOS SDN controller. However, authors in [15] , [22] claim that the different controllers can have different security modeling, and therefore, different countermeasures.

SDN can be deployed in different network scales [7].

To generate more intrinsic data for SDN networks, the network topology should be created using physical devices. A more intrinsic dataset can be generated using physical topology with many connected devices.

Ideally, obtaining network traffic data from global SDN deployments such as Google B4 and cloud computing infrastructures will make it possible for the research community to examine the pecularities of Intrusion Detection reseach in SDN.

5 Inference of DDoS Attacks

To identify DDoS attack traffic in the flow-level traces provided by the InSDN dataset, we employ a deep learning approach.We also aim to perform denoising and dimensionality reduction where appropriate.

5.1 Data Pre-Processing

Socket information such as Source IP, Destination IP, and flow ID are removed to avoid the overfitting problem because these features are different from network to network. The final dataset includes 77 various features, besides the traffic category (i.e. the Label assigned to each flow instance in our Supervised Learning model) [7]. In order to facilitate binary classification, We use one-hot encoding on the label values where, the strings Normal and Attack are encoded into binary values of 0 and 1, respectively. The InSDN dataset also includes as many as 8 zero variance features that do not contain any information useful for classification purposes. These features such as 'Fwd Byts/b Avg' have either one unique value (zero variance) or a high ratio of the most common value to the next most common value therefore are removed. Finally TCP flags are removed leaving us with a total of 52 numerical features that have different ranges, hence standardized to the scale of the values between -1 and 1.

5.2 Data Analysis

Our featureset includes basic statistics like the duration of a flow and flow Inter-arrival time as well as SDN-specific features such as Min, Max, and Standard deviation of these statistics. First, we compute the pair-wise correlation matrix for all the features and observe where they approach thresholds of 0.7, 0.8, and 0.9. As indicated by Figure 2, the

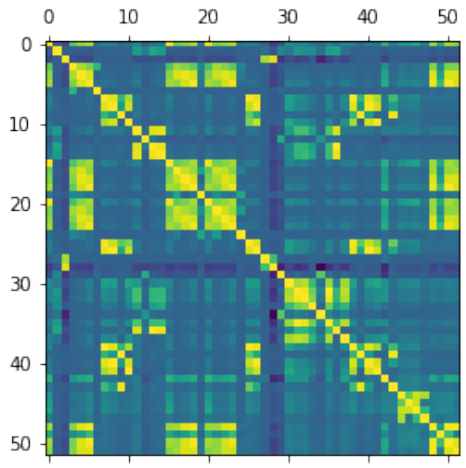

Figure 2 – Pair-wise Correlation Matrix of the features.

correlation matrix reveals that time-related features such as Flow Duration,Flow Inter-Arrival Time,Idle Std seem to be correlated. Moreover, it can be observed that SDN specific features such as maximum, minimum, mean, and standard deviation of time-based network values are also correlated. These SDN features can be directly extracted from the network controller through API queries. Next we reduce the dimensionality of the feature

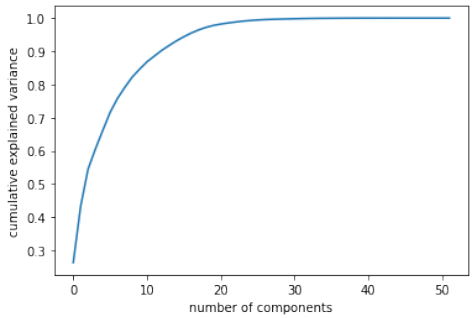

Figure 3 – Cumulative Explained Variance by Principal Components.

space by applying a principle component analysis (PCA). A PCA decomposition can be used to project a high-dimentsional space to a lower-dimensional space by relying the on the initial principal components. In effect, it converts a set of values of M possibly correlated variables into a set of K uncorrelated variables, the PCAs. in that regard, PCA is a clustering algorithim for high-dimensional data. we find that a significant number of our features are correlated since the first 5 PCAs explain about 70% of the variance (refer to Figure 3) and the first 12 more than 90%. Figure 4 shows the projection of our feature space into the first 2 PCAs. We can see in Figure 4 that Attack and Normal traffic are to a certain extent visually separable.

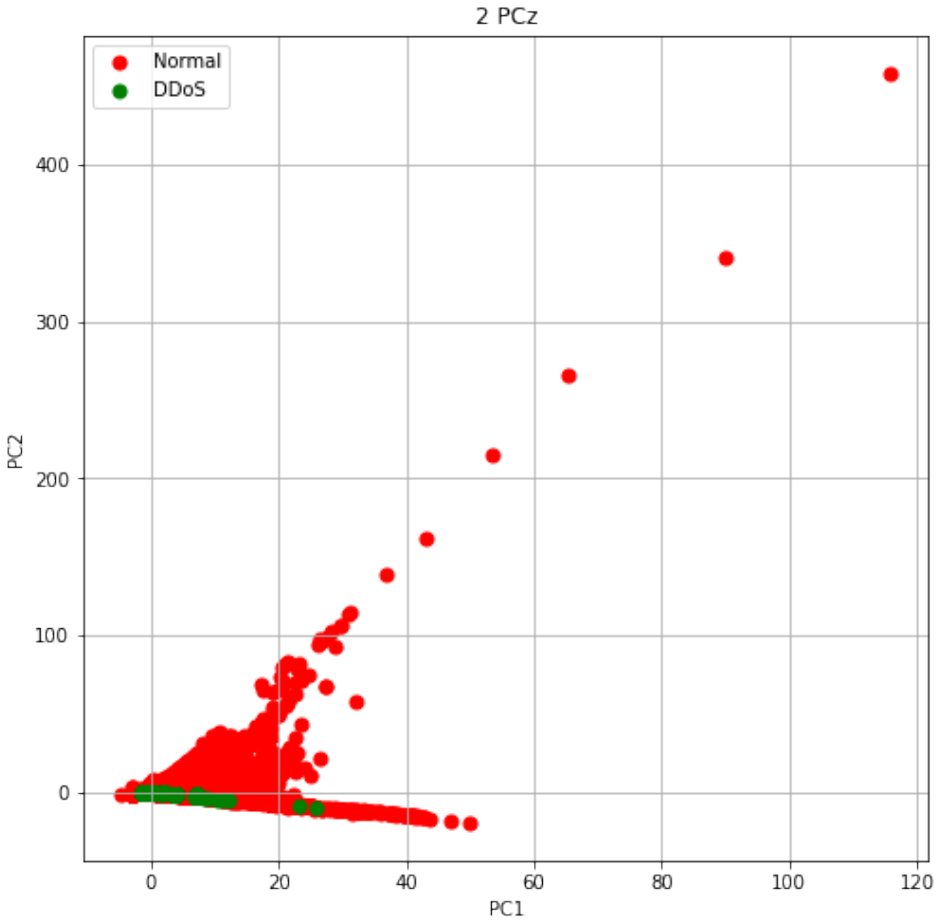

Figure 4 – projection of feature space to the first 2 PCAs.

6 Workflow, Training, and Testing

Deep learning allows computational models composed of many layers to learn the data structure with many levels of abstraction. In this study, we seek to design, implement, and evaluate neural network classifiers that learn the spatial and temporal structure of Attack traffic as opposed to Normal traffic. We propose two different deep learning models that achieve remarkable results and later demonstrate that the number of false negatives in our detection scheme drops drastically when these models cooperate.

6.1 Experimental Setup

The experiments were carried out using Python and the Pytorch [21] library. Table 1 details the system specifications.

Table 1 – System Specifications

Unit	Description
Processor	2199.998 MHz
Memory	13G
L3 cache	56320K

6.2 Data Splitting

In machine learning, one of the main requirements is to build computational models with high prediction and generalization capabilities. In the case of supervised learning, a computational model is trained to predict outputs of an unknown target function. The target function is represented by a finite training dataset T of examples of inputs and the corresponding desired outputs.At the end of the training process, the final model should predict correct outputs for the input samples from T, but it should also be able to generalize well to previously unseen data [23]. We split the 141953 network flow traces provided by the InSDN dataset for DDoS traffic classification into the training and test sets with a 80% train, 20% test ratio.

6.3 Classification with Raw Features

Fully Connected Neural Networks are great at classification. Our intention is to gain multiple layers of abstraction and we reckon that a 7 Layer Perceptron as depicted in Figure 5 with 52, 128, 512, 512, 128, 64, and 16 nodes in each layer can help us expand across the data space and ultimately reduce the dimentionality. The rectified linear activation function (ReLU) has been used at each layer to increase the non-linearity degree and set all negative values in the feature set by zero. Finally, the output layer incorporates the Sigmoid function to represent the probability of each input flow belonging to either class. We use the Adam optimization algorithm for stochastic gradient descent where the learning rate set to 0.0001 and train the deep learning model. At each backpropagation step during the training process, We calculate the reconstruction loss using the Binary Cross Entropy criterion.

6.4 Denoising via Convolutional AutoEncoder

The Autoencoder, generally deployed as a generative model, is proficient at extracting high-level features from the data by transforming it into a latent space. The latent space

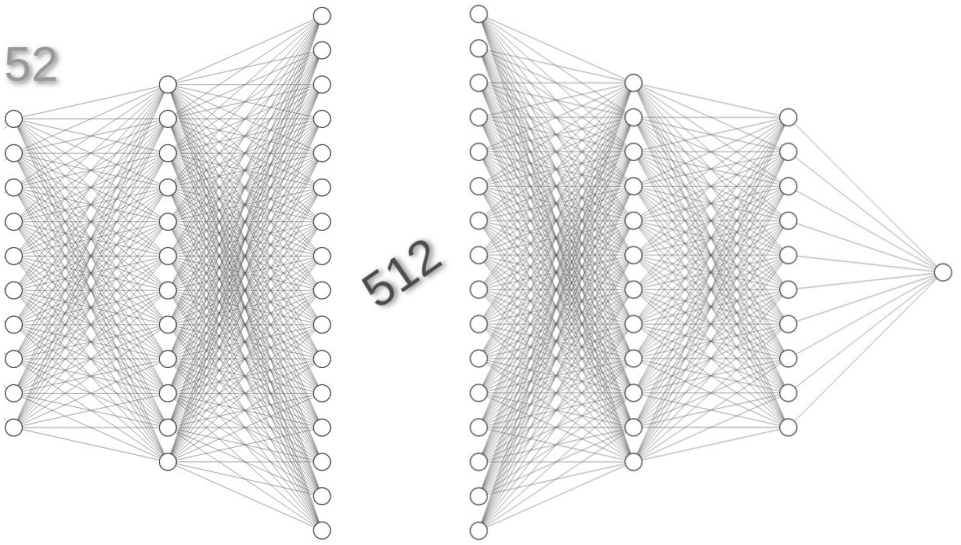

Figure 5 – Fully Connected Neural Network

view representation of the data lies at the bottleneck layer of the Autoencoder which is later used to generate new data instances in image processing applications. Convolutional Autoencoders have proven to be excellent at denoising data. For this reason, we set out to design a Convolutional Autoencoder and train it over the InSDN dataset. The bottleneck representation of the dataset can reveal quite a bit about the spatial and temporal structure of the dataset which we use as a feature extraction step in the deep learning process.

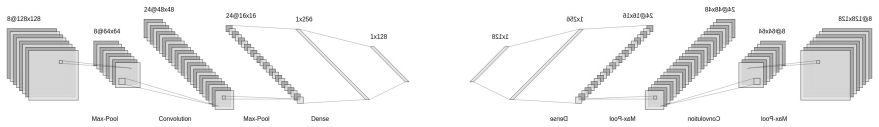

Figure 6 – Convolutional Autoencoder

As a first pre-processing step, we reshape each flow instance of dimensions [1,52] into an image-like structure of shape [8,8]. This transformation is carried out using a Linear transformation from 52 to 64 features which are then cast to the (1,8,8) Tensor shape in order to suit the input dimensions of the convolutional autoencoder. Generally speaking, Autoencoders are made up of the encoder, the bottleneck, and the decoder. The architecture of the convolutional autoencoder (depicted in Figure 6) is composed of three convolutional layers taking in flow-level values through one input channel and expanding it first to 8 and next to 16 and 32 channels. Each channel corresponds to a different filter applied throughout the convolution process. The output of the convolutional layers is then flattened and linearly transformed into the bottleneck space dimension of [1,4]. The Decoder has the same architecture as the encoder except in reverse order. throughout the training process, the decoder reconstructs training data instances and we can measure

and minimize loss.

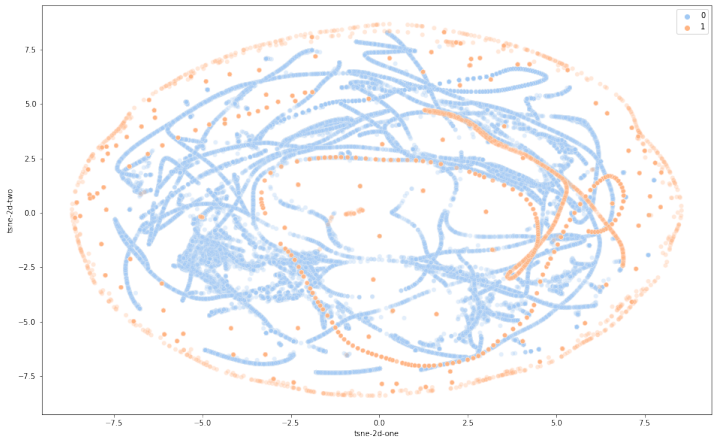

Figure 7 – t-Distributed Stochastic Neighbor embedding analysis with Raw features

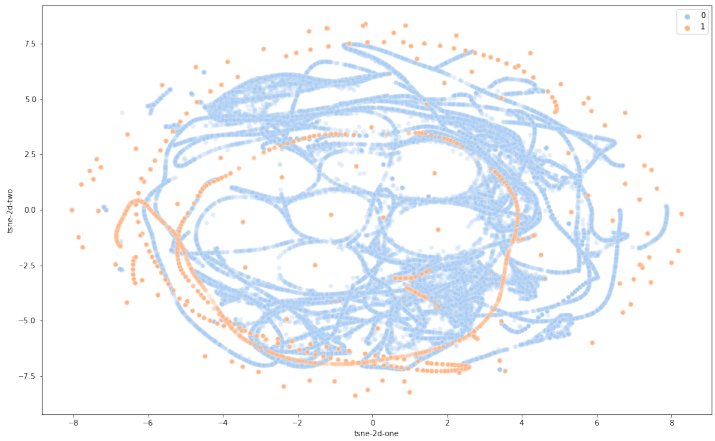

Figure 8 – t-Distributed Stochastic Neighbor embedding analysis with High-level features

The convolutional autoencoder is trained over the train set for a total of 200 epochs leaving us with a latent space representation view of the network traffic data. In this fashion, we can extract 4 high-level features and have the dimension of our featureset reduced from 52 to 4. The choice to go with 4 high-level features was made after experimenting with other candidates such as 8, 16, and 32. Our experiment demonstrates that 4 high-level features achieve the best classification results. As depicted in Figure **??**, our t-distributed stochastic neighbor embedding analysis reveals that the convolutional autoencoder successfully denoises the flow data making it easier for a classifer to detect outliers as DDoS instaces seem to be closer together.

6.5 Classification with High-Level Features

By extracting high-level features using a Convolutional Autoencoder in the previous section, we were able to transform the data into four-dimensional space. Next, we design,

train, and evaluate a fully connected neural network for classification. Therfore, we insert 4-dimensional output from the bottleneck of the convolutional autoencoder into a minimal classifer of 5 layers. Figure 9 shows the the architecture of fully connected classifer consisting of four linear layers going from 4 -> 16 -> 32 -> 32 -> 16 -> 1 nodes at each layer with the rectified linear unit as the activation function.

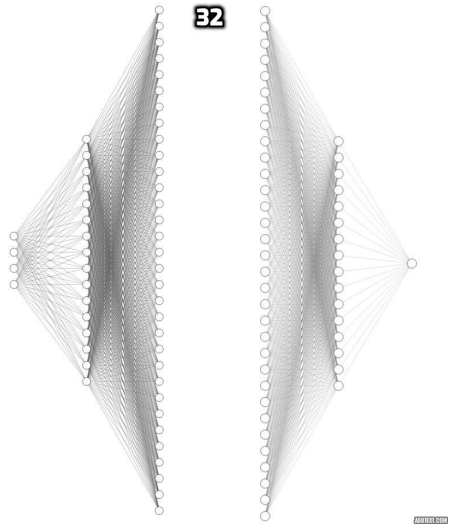

Figure 9 – Fully Connected Neural Network

7 Experimenting with Other Attack Types

To further evaluate the performance of our classification models, we set out to experiment with the entire InSDN dataset that also includes instances of Probe and DoS attacks as well as a few instances of BFA and U2R attacks. First, we formulate this experiment a a binary classification problem where flow instances are classified into the Normal and Intrusion classes. These labels have been encoded to 0 and 1, respectively. After splitting the dataset into train and test sets with a 80% and 20% ratio, we train the Convolutional Autoencoder **??** for a total of 200 epochs over train data.

Next, we calculate the latent space representation of the train data by feeding it through the Autoencoder. Said high-level representation of data lies at the bottleneck layer of the autoencoder and is measured in 4 dimensions. Figures 10 and 11 show the distribution of Raw features vs high-level features in the data space. The comparison between these two figures demonstrates that the Autoencoder was able to accomplish the denoising of the data. It can be observed on the t-SNE plots that high-level features lie closer together with respect to distances between them. The Relevant information is in the relative distances between low dimensional points. t-SNE captures structure in the sense that neighboring points in the input space will tend to be neighbors in the low dimensional space.

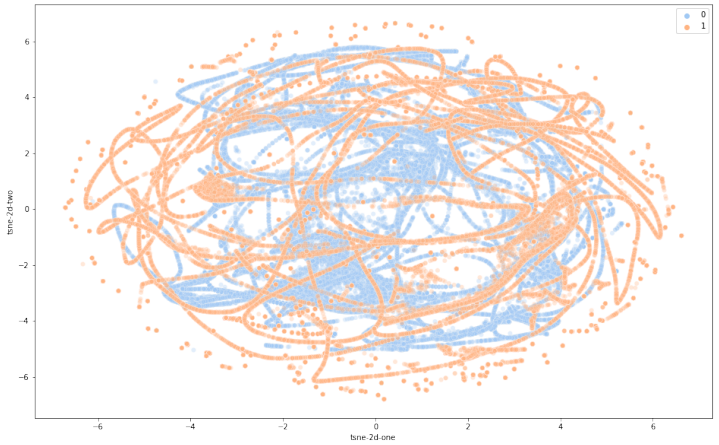

Figure 10 – t-SNE analysis of high-level features

Next, we train a fully connected classifier over the high-level features for a total of 4000 epochs. Figure 9 shows the architecture of the fully connected classifier.

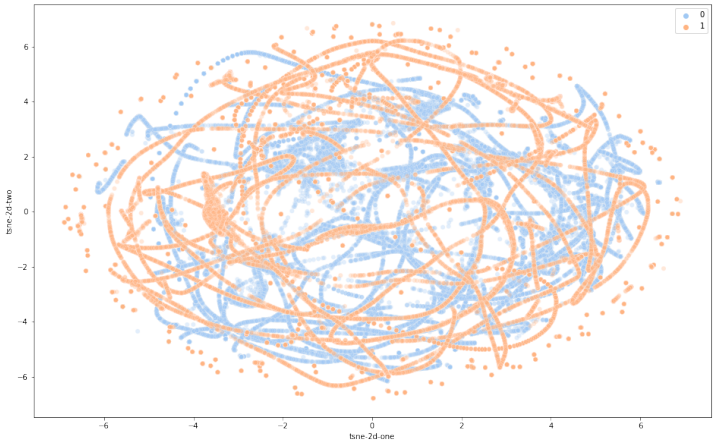

Figure 11 – t-SNE analysis of high-level features

8 Evaluation Metrics

To evaluate the performance of our classifiers, we use the classification accuracy, precision, recall, and F1 score as performance metrics. In addition, we calculate the confusion matrix where:

- `True Positive (TP)` indicates attack traffic correctly classified as malicious(DDoS).
- `True Negative (TN)` indicates normal traffic correctly classified as benign.
- `False Positive (FP)` indicates normal traffic incorrectly classified as malicious.
- `False Negative (FN)` indicates attack traffic incorrectly classified as normal.

9 Results

9.1 Results for Classification with Raw features

The classifier is trained over the train set in the course of 200 epochs and demonstrates a remarkable accuracy of 99.60% with the training loss dropping drastically a quarter into the training process. Table 2 reports the Accuracy, Precision, Recall, and F1 metrics for the performance of our classification network. Figure 12 shows the training loss at each epoch. We can see that the training loss drops drastically half-way through the training process and our model's performance converges after that.

Table 2 – Classification Performance

Accuracy(%)	Precision(%)	Recall(%)	F1-Score(%)
99.60	99.27	99.97	99.62

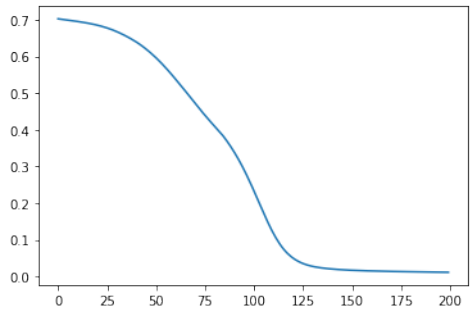

Figure 12 – Training Loss in Classification with Raw features

9.2 False Negatives

In the context of DDoS detection, a false negative means an attack instance was missed by our classifier. While Our fully connected classifier performs remarkable in detecting DDoS vs normal traffic, there are as many as 4 false negatives in the confusion matrix (Figure 13).

9.3 Results for Classification with High-level Features

we insert 4-dimensional output from the bottleneck of the convolutional autoencoder into the fully connected classifer and train it over the train dataset for a total of 4000 epochs. Table 3 reports the remarkable performance metrics achieved by our classifer.

Table 3 – Classification Performance

Accuracy(%)	Precision(%)	Recall(%)	F1-Score(%)
99.28	99.22	99.40	99.31

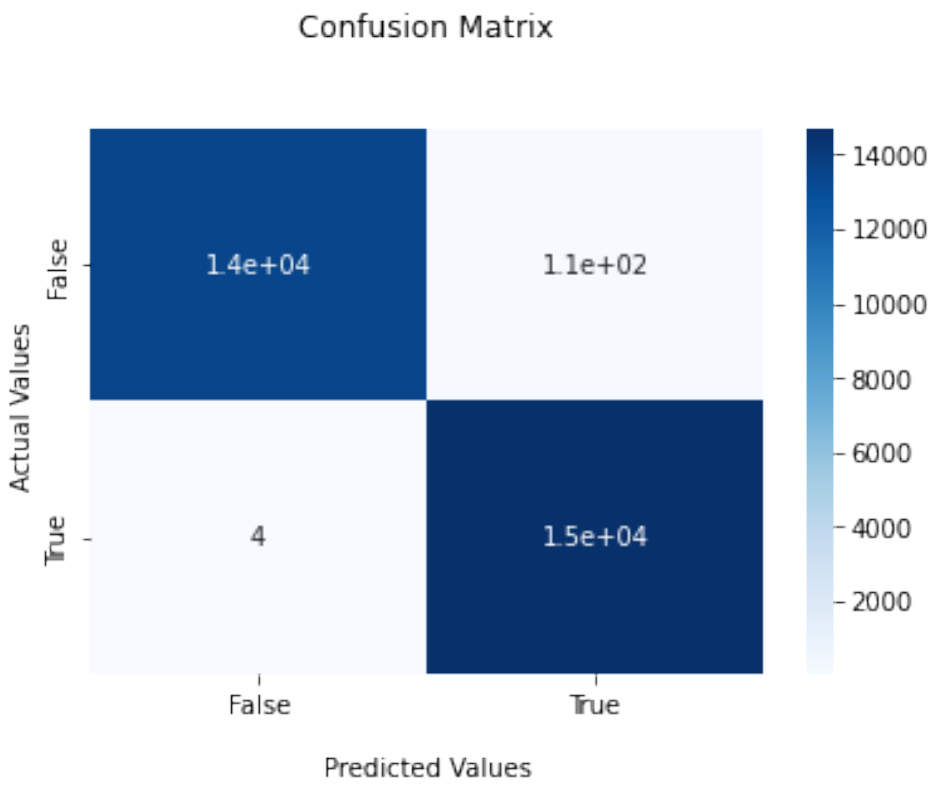

Figure 13 – Confusion Matrix

9.4 Results for Cooperative DDoS Detection

In the previous section, we proposed two models for DDoS detection in Software-Defined Networks based on the InSDN dataset. Both models perform remarkably with accuracy rates in the 99th percentile. we remark and here demonstrate that the number of false negatives in DDoS detection drastically decreases when models cooperate.

In order to minimize the number of false negatives in our DDoS detection scheme, we combine the proposed classification models in such a fasion that they share their decisions on each flow instance. Table 4 reports the performance metrics of our proposed cooperative DDoS detection scheme. As indicated by the Confusion Matrix (Figure 17), the number of false negatives drops from 4 in the first model to 1 in the cooperative model.

Table 4 – Performance of Cooperative DDoS Detection

Accuracy(%)	Precision(%)	Recall(%)	F1-Score(%)
99.57	99.18	99.99	99.58

Best results are achieved through cooperation of IDS applications whether they're in different locations across the network as in the case of software-defined networks with multiple controllers or placed on the same central point on top of the controller. It's when these model cooperate and exchange data that we can ensure a better security posture and threat mitigation strategies.

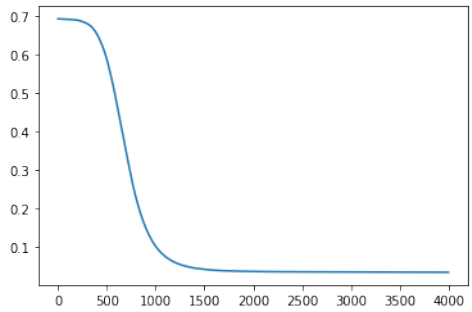

Figure 14 – Training Loss for Classification with high-level features

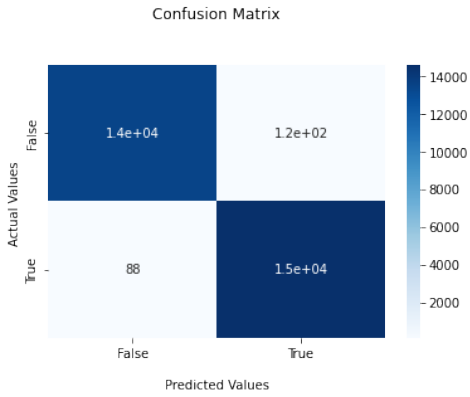

Figure 15 – Confusion Matrix for Classification with high-level features

9.5 Results for Cooperative Intrusion Detection

The performance metrics are reported in Table 5. Even though the proposed model acheives a remarkable accuracy of 99.06%, the Confusion Matrix (Figure 18) begs the question: "How can we drop the number of false negatives in this detection scheme ?". Moreover, Figures 19 , 20 and Table 6 show the misclassified ratios under their respective attack class. It can be deduced from the class distribution of the misclassified instances that our proposed deep learning model gravitates towards detection of minority classes as opposed to majority classes such as DDoS and DoS.

Table 5 – Classification Performance (with High-level Features)

Accuracy(%)	Precision(%)	Recall(%)	F1-Score(%)
99.06	99.15	99.45	99.30

Table 6 – Ratio of Misclassified Instances under their respective class

BFA	DDoS	DoS	Normal	Probe	U2R
0	0.0016	0.2026	0.0004	0.00004	0.0588

As discussed in the previous sections, The lesson learned from Cooperative DDoS de-

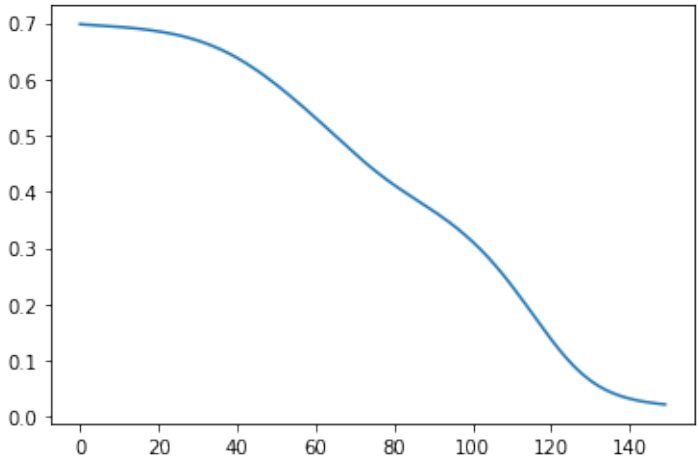

Figure 16 – Training Loss of Cooperative DDoS Detection

tection was that the number of false negatives drop drastically when models cooperate. By the same logic, We train a fully connected classifer (Figure 21) over raw features. The performance results from this cooperative detection are reported in Table 7 . Figure 22 shows the confusion matrix. Our analysis tells us as many as 44 flow instances were detected only through cooperation.

Table 7 – Classification Performance (with Raw Features)

Accuracy(%)	Precision(%)	Recall(%)	F1-Score(%)
99.41	99.13	99.99	99.56

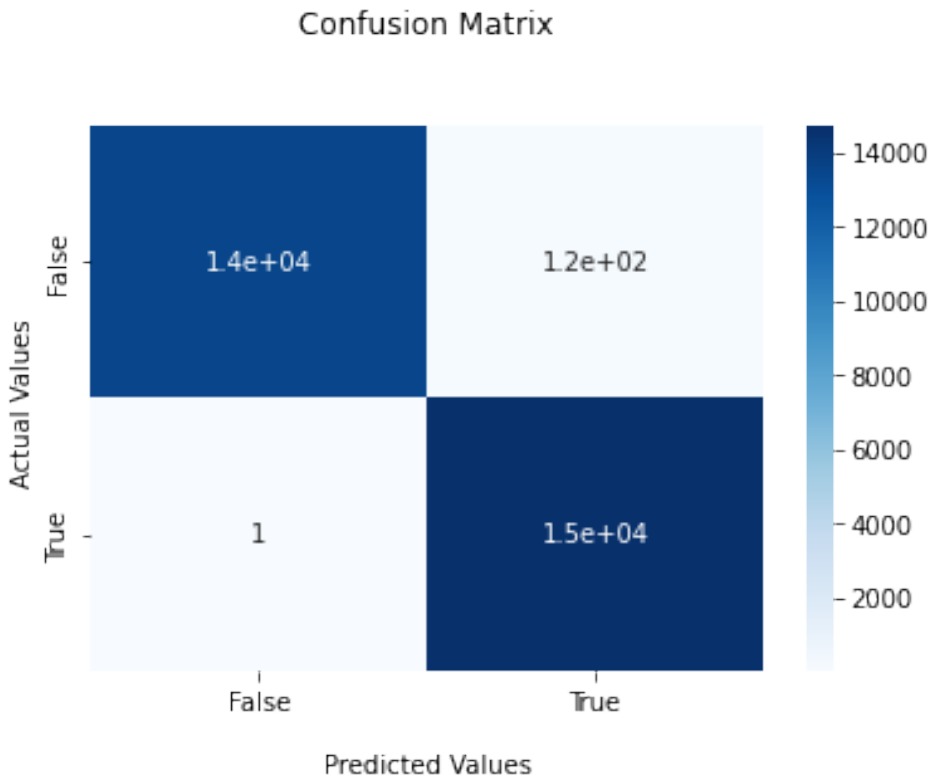

Figure 17 – Confusion Matrix of Cooperative DDoS Detection

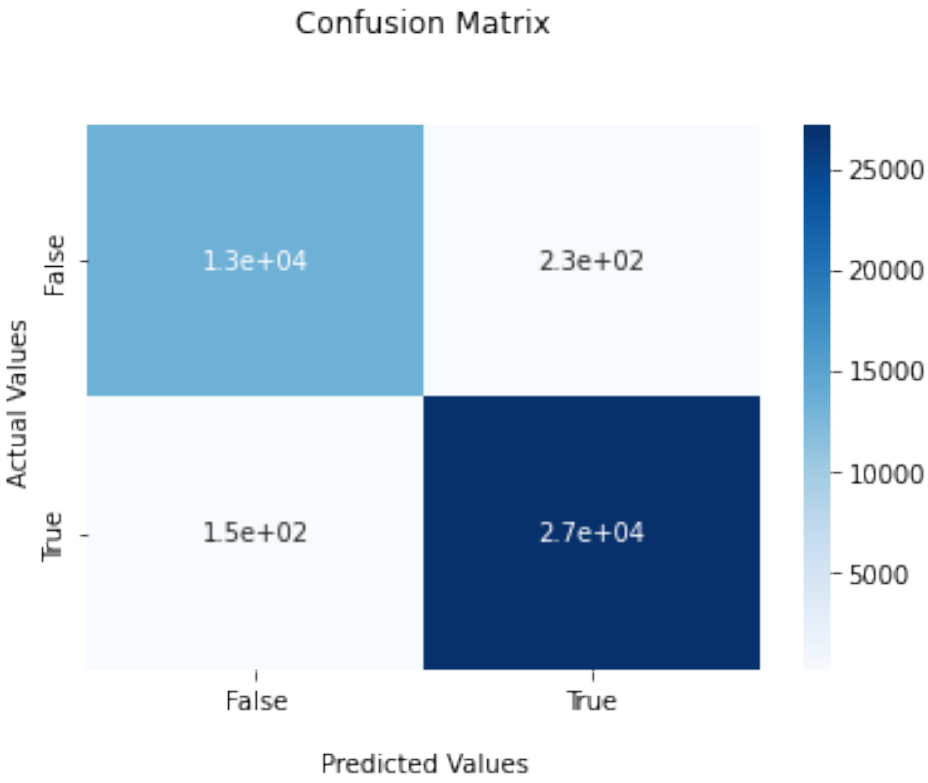

Figure 18 – Confusion Matrix (Classification with High-level features)

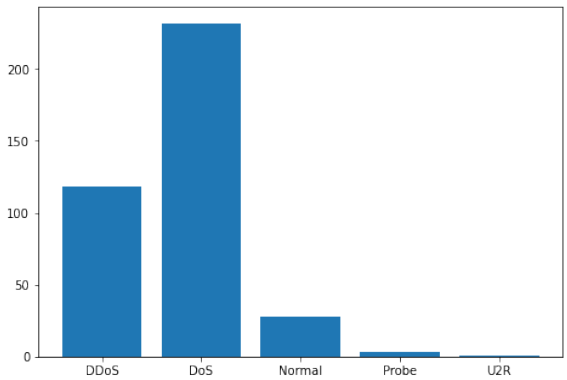

Figure 19 – Misclassified Flow Instances under their respective classes)

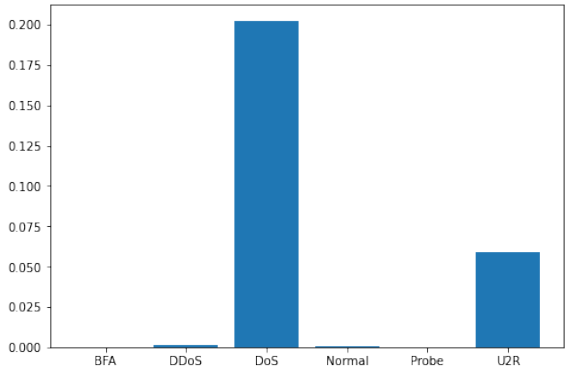

Figure 20 – Ratio of Misclassified Instances under their respective class

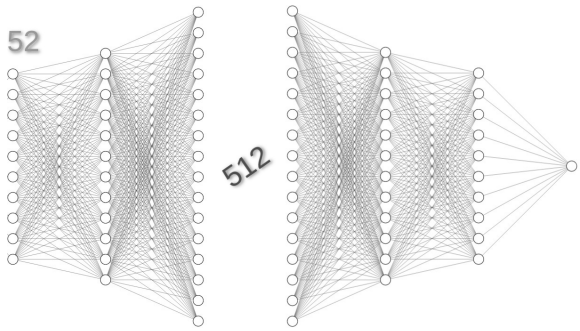

Figure 21 – Fully Connected Classifier

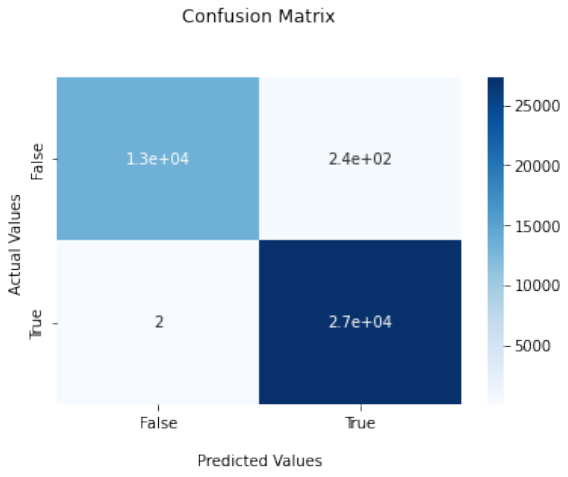

Figure 22 – Confusion Matrix from Cooperative Intrusion Detection

10 Discussion and Future Work

In this study, we walked through the specifics of DDoS attacks and how we can detect instances of such attacks in network traffic data within an SDN enviroment. Morover, we laid out the pecularities of cyber threat intelligence in software-defined networks and made the assertion that these traits call for a fresh perspective on detection of DDoS attacks in future networks. While this exemplary look into DDoS attacks in SDNs as provided by the InSDN dataset reveals a lot about the scientific approach to the problem, there are aspects to the subject that best be addressed:

- The InSDN dataset, while providing the first novel SDN intrusion detection dataset, only includes synthetic DDoS attacks generated in the SDN testbed characterized by its authors. We understand that the quality of a good model depends on the quality of the dataset and how well it measures the phenomenon under study. with this in mind, we need more data from software-defined networking deployments. Google's b4 ,Espresso are good examples. cloud computing infrastructures can also provide us with network data of real ddos instances versus normal traffic.

- Many catastrophic real-world cases of DDoS attacks incorporate amplification attacks and reflector elements across the network. Therefore, a future study may zero in on instances of DDoS Amplification attacks in SDN as they generate orders of magnitude more traffic than mere flooding attacks. Moreover, the attacker may take advantage of the centralized SDN architecture and target a network service as reflector within a DDoS amplification attack scheme.

- In discussing how to detect attacks within network traffic data, one can measure spatial features of network traffic data whereas it's equally important, if not more, to take into account the temporal features of attacks. this notion is even more crucial in the case of DDoS attackS where it's usually not any one network flow that helps identify an attack but sequences of them as DDoS attacks take place over the course of consecutive periods. Therefore, a potential angle would be to formulate the problem as a sequence modeling problem. LSTM units in Neural Networks are a great choice of model for such an approach.

- Although SDN is deployed in different network environments, It is still evolving. The previous history of SDN attacks is unknown. Therefore, the authors of [7] act like the attacker and anticipate the weaknesses that he might be likely to strike.

11 Conclusion

DDoS attacks were first observed twenty years ago, but they are still one of the most serious threats. The hallmarks of Software-defined networking call for a fresh perspective on DDoS detection in SDN. The InSDN dataset paves the way for the research community to take a fresh look at DDoS attacks in SDN. However, we need more data from SDN deployments that shed a light on the pecularities of DDoS attacks in SDN. In this study, we proposed two deep learning models that achieve excellent performance metrics. Next, we demonstrated that a better DDoS detection scheme can be proposed when models cooperate. Our results exhibit a drastic fall in the number of false negative when models cooperate.

List of Figures

List of Tables

References

[1] M. Abdallah, N. An Le Khac, H. Jahromi, and A. Delia Jurcut. A hybrid cnn-lstm based approach for anomaly detection systems in sdns. In *The 16th International Conference on Availability, Reliability and Security*, pages 1–7, 2021.

[2] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis, et al. Understanding the mirai botnet. In *26th USENIX security symposium (USENIX Security 17)*, pages 1093–1110, 2017.

[3] S. Behal and K. Kumar. Trends in validation of ddos research. *Procedia Computer Science*, 85:7–15, 2016.

[4] Y. Bengio, Y. LeCun, et al. Scaling learning algorithms towards ai. *Large-scale kernel machines*, 34(5):1–41, 2007.

[5] O. Çetin, C. Ganán, L. Altena, T. Kasama, D. Inoue, K. Tamiya, Y. Tie, K. Yoshioka, and M. Van Eeten. Cleaning up the internet of evil things: Real-world evidence on isp and consumer efforts to remove mirai. In *NDSS*, 2019.

[6] A. Divekar, M. Parekh, V. Savla, R. Mishra, and M. Shirole. Benchmarking datasets for anomaly-based network intrusion detection: Kdd cup 99 alternatives. In *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, pages 1–8. IEEE, 2018.

[7] M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut. Insdn: A novel sdn intrusion dataset. *IEEE Access*, 8:165263–165284, 2020.

[8] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred. Statistical approaches to ddos attack detection and response. In *Proceedings DARPA information survivability conference and exposition*, volume 1, pages 303–314. IEEE, 2003.

[9] Z. Guangsen and M. Parashar. *Journal of Research and Practice in IT*, 38(1):69–84, 2006.

[10] A. Guerra-Manzanares, H. Bahsi, and S. Nõmm. Hybrid feature selection models for machine learning based botnet detection in iot networks. In *2019 International Conference on Cyberworlds (CW)*, pages 324–327, 2019.

[11] S. Jain, A. Kumar, S. Mandal, J. Ong, L. Poutievski, A. Singh, S. Venkata, J. Wanderer, J. Zhou, M. Zhu, et al. B4: Experience with a globally-deployed software defined wan. *ACM SIGCOMM Computer Communication Review*, 43(4):3–14, 2013.

[12] C. C. JC, J. Imbachi, and B. V. JF. Security in sdn: A comprehensive survey. 2020.

[13] M. Jonker, A. King, J. Krupp, C. Rossow, A. Sperotto, and A. Dainotti. Millions of targets under attack: a macroscopic characterization of the dos ecosystem. In *Proceedings of the 2017 Internet Measurement Conference*, pages 100–113, 2017.

[14] M. Jonker, A. Sperotto, R. van Rijswijk-Deij, R. Sadre, and A. Pras. Measuring the adoption of ddos protection services. In *Proceedings of the 2016 Internet Measurement Conference*, pages 279–285, 2016.

[15] R. Khondoker, A. Zaalouk, R. Marx, and K. Bayarou. Feature-based comparison and selection of software defined networking (sdn) controllers. In *2014 world congress on computer applications and information systems (WCCAIS)*, pages 1–7. IEEE, 2014.

[16] D. Kopp, C. Dietzel, and O. Hohlfeld. Ddos never dies? an ixp perspective on ddos amplification attacks. In *International Conference on Passive and Active Network Measurement*, pages 284–301. Springer, 2021.

[17] D. Kreutz, F. M. Ramos, and P. Verissimo. Towards secure and dependable software-defined networks. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 55–60, 2013.

[18] Z. M. Mao, V. Sekar, O. Spatscheck, J. Van Der Merwe, and R. Vasudevan. Analyzing large ddos attacks using multiple data sources. In *Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense*, pages 161–168, 2006.

[19] S. Nõmm and H. Bahşi. Unsupervised anomaly based botnet detection in iot networks. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 1048–1053, 2018.

[20] E. Osterweil, A. Stavrou, and L. Zhang. 21 years of distributed denial-of service: Current state of affairs. *Computer*, 53(7):88–92, 2020.

[21] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[22] K. Phemius, M. Bouet, and J. Leguay. Disco: Distributed multi-domain sdn controllers. In *2014 IEEE Network Operations and Management Symposium (NOMS)*, pages 1–4. IEEE, 2014.

[23] Z. Reitermanova et al. Data splitting. In *WDS*, volume 10, pages 31–36, 2010.

[24] S. Scott-Hayward, G. O'Callaghan, and S. Sezer. Sdn security: A survey. In *2013 IEEE SDN For Future Networks and Services (SDN4FNS)*, pages 1–7. IEEE, 2013.

[25] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani. A detailed analysis of the kdd cup 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*, pages 1–6. Ieee, 2009.

[26] Y. Wei, J. Jang-Jaccard, F. Sabrina, A. Singh, W. Xu, and S. Camtepe. Ae-mlp: A hybrid deep learning approach for ddos detection and classification. *IEEE Access*, 9:146810–146821, 2021.

[27] Q. Yan, F. R. Yu, Q. Gong, and J. Li. Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges. *IEEE communications surveys & tutorials*, 18(1):602–622, 2015.

[28] K.-K. Yap, M. Motiwala, J. Rahe, S. Padgett, M. Holliman, G. Baldus, M. Hines, T. Kim, A. Narayanan, A. Jain, et al. Taking the edge off with espresso: Scale, reliability and programmability for global internet peering. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 432–445, 2017.

[29] S. Yu, J. Zhang, J. Liu, X. Zhang, Y. Li, and T. Xu. A cooperative ddos attack detection scheme based on entropy and ensemble learning in sdn. *EURASIP Journal on Wireless Communications and Networking*, 2021(1):1–21, 2021.