5-2023

Deep Learning Enhanced Visulization Tool For Network Monitroing

Vaishali Bharatrao Dhemare

DEEP LEARNING ENHANCED VISULIZATION TOOL FOR NETWORK MONITROING

A Thesis

by

VAISHALI BHARATRAO DHEMARE

Submitted to the Office of Graduate Studies of
Prairie View A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

May 2023

Major Subject: Electrical Engineering

DEEP LEARNING ENHANCED VISULIZATION TOOL FOR NETWORK MONITROING

A Thesis

by

VAISHALI BHARATRAO DHEMARE

Submitted to the Office of Graduate Studies of
Prairie View A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved as to style and content by:

Dr. Lijun Qian
(Chair of Committee)

Dr. Xiangfang Li
(Committee Member)

Dr. Xishuang Dong
(Committee Member)

Dr. Pamela Obiomon
(Committee Member)

Dr. Pamela Obiomon
(Department Head)

Dr. Pamela Obiomon
(Dean of Graduate School)

Dr. Dorie J. Gilbert
(Dean of Graduate School)

December 2022

Major Subject: Electrical Engineering

ABSTRACT

Deep Learning Enhanced Visualization Tool for Network Monitoring

(December 2022)

Vaishali B. Dhemare, B.E., PVPIT College of Engineering Shivaji University,

M.S., Prairie View A&M University

Chair of Advisory Committee: Dr. Lijun Qian

In this era of web technology driven by social networks, cloud computing, big data, and E-business, technology is also rapidly evolving. Most of the information is stored and managed via the internet. With an increase in these development tools and techniques, cyber-crime is constantly increasing. The level of damage these attacks cause to the system affects the organizations to the core. Contemporary Deep Learning and Machine Learning technologies have become the popular choice of intrusion detection systems for the detection and prediction of cyber-attack. Similarly, cyber-security visualization is also an integral and essential part of monitoring network traffic and optimization. Abundant work has already been done to detect attacks, but monitoring these attacks still appears as elusive as detection for cyber analysts. However, the current open-source visualization tool has not been integrated with Deep Learning models to gain intelligence on the network. While many researchers [3] are already working on cyber-attack defense mechanisms, this research also takes advantage of Deep Learning and Machine Learning technologies to contribute to the work against such crimes. A novel Deep Learning enhanced visualization tool is also proposed for malicious traffic node prediction and monitoring. The proposed method exploits the intriguing properties of Deep Learning models to gain intelligence for network monitoring. A real-world DARPA dataset has been used to validate the proposed method.

***Index Terms*—Cyber-security, data analysis, data science, darpa-dataset, decision tree, deep learning, deep neural network, DL model, ML model, network analysis tool, network monitoring tool, supervised learning, support vector machine, visualization tool.**

DEDICATION

To

My Family & Friends

And

Prairie View A&M University

ACKNOWLEDGEMENTS

This endeavor would not have been possible without the grace of God Almighty, who has been giving me strength through this process. I sincerely appreciate the elegance and privilege I have enjoyed in this journey and throughout my master's program. This process of working on my thesis has been an incredible and adventurous journey-where I have shaped my future for betterment.

I could not have undertaken this journey without a few people, and I want to take advantage of this moment to thank those who were part of this voyage.

Firstly, I would like to thank my supervisor and advisor Dr. Lijun Qian, for your guidance in choosing the right direction and staunch support during the journey. This would not have been possible.

My sincere thanks to Dr. Xishuang Dong and Dr. Xiangfang Li, Assistant Professors in the Electrical Engineering department. Your profound knowledge, advice, and provided tools during all my master's coursework have helped me make this project a success.

I am grateful to all the members of the CREDIT (Center of Excellence in Research and Education for Big Military Data Intelligence) Center for the timely help and association that has made me achieve my goal in Prairie view. Worlds cannot express my gratitude towards the CREDIT Center team.

Finally, I appreciate the immense support from friends and family. Thank you to my lovely husband and daughter for the encouragement and prayers. My special appreciation to my friends and classmates. Without your profound and timely support, this would not have been possible. I am genuinely grateful for all the help.

This research work is partly supported by the United States Office of the Under Secretary of Defense for Research and Engineering (OUSD (R&E)) under agreement number FA8750-15- 2-0119. The United States Government is authorized to reproduce and distribute reprints for governmental purposes, notwithstanding any copyright notation thereon.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

I. INTRODUCTION

Background of Cyber-security for Network Monitoring

The role of the computer and the internet in this digital era is well recognized. Recent developments in cyberspace and networking have greatly benefited societies. Still, the explosive growth of cyberspace has also led to unethical practices by individuals who use technology to exploit others. Such exploitation of tampering with secure information and data, disabling networks, stealing money, and spying is known as cyber-attacks. There are many ways cyber attackers can harm the system and network, which could collapse organizations to their core. Along with the new inventions, there is a need to have appropriate defense measures against such attacks. With the rapidly increasing number of cyber-crimes, there has been a dearth of knowledge about these attacks, rendering many individuals/agencies/organizations vulnerable. [3] There are many researchers already working on defense mechanisms. This research work took advantage of Deep Learning and Machine Learning technologies to contribute to the work against such cyber-crime. This research the prediction of cyber-attacks on the IP network by using existing data of such attacks. The visualization tool integrated with the backend system made monitoring these attacks easy and the user can take the appropriate actions to avoid further attacks and damage to the networks.

Cyber-attack on IP Network

Today's world is undeniably dominated by technology; this fact is well known to all. New technologies have been developed to improve lifestyles [5]. The role of the internet in modern society is inevitable. Cyberspace is a vast terminology. In simple words cyberspace is defined as, "the virtual area created for communication using wires or fiber

This thesis style is in accordance with IEEE transactions on Cyber security and Deep Learning.

optic cables to transmit information to and from the internet." This space has been increasing gradually in size as more information is fed into it. Cyberspace has gradually pervaded all aspects of human life, such as banking, hospitals, education, emergency services and the military. Its complexity has also been increasing [6] along with the threats. Such threats are called cyber-attacks.

A typical IP network [1] consists of different network elements like LANs, switches, routers, computers, and firewalls, as shown in Fig. 1.1. Similarly, attackers also connect to the network through wired or wireless connections. Even though the network is secured with firewalls and DMZ techniques, network attackers still try to bypass security mechanisms by exploiting the vulnerabilities of the target network. DMZ is a type of subnetwork placed between the private network and public internet to protect and assign an additional layer of security to organizations to handle untrusted IP traffic.

Network attacks disrupt the network operations, and include:

- Overloading a network and denying legitimate IP traffic for users.
- Malfunctioning network devices.
- Reducing network throughput.
- Many more such activities.

An attacker takes advantage of misconfigurations in the software services and exploits the system's loopholes and bugs to disturb normal network activities. For launching the attack, attackers intentionally identify the loopholes based on standard services open on a host, gather relevant information, and misuse it [1].

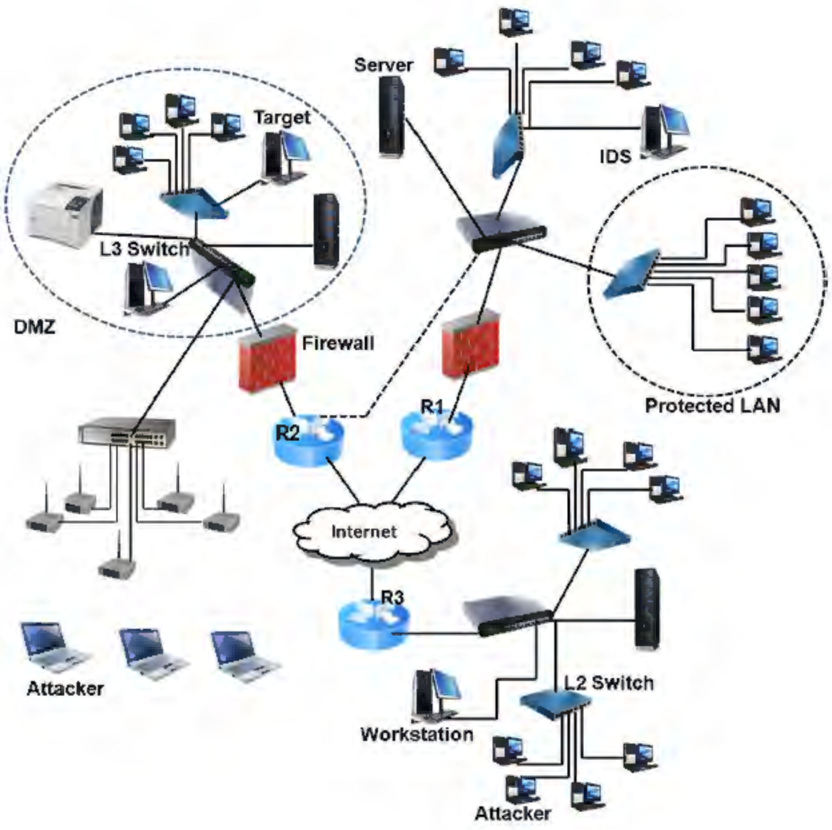


Fig. 1.1 Typical network structure with protected LAN, DMZ, and IDS deployment [1].

Types of Cyber-attack and need of Defense Mechanisms

The nature, severity, and complexity of these cyber-attacks are increasing over time [7]. Lack of understanding of the various types of attacks, their severity, and their mode of spreading has made many organizations vulnerable. Developing security measures requires a thorough understanding of cyber-attacks and their classifications. Many findings have already been made so far and reach is a concern, but as time passes, the nature and types of these attacks are changing with changing technologies. Organizations need to keep evolving their defense mechanism as the nature and approach of attackers become more sophisticated with time. Therefore, a comprehensive listing of cyber-attacks and classifications of attacks form a vital constituent of cyber-security initiatives. These studies on classification based on characteristics such as purpose, severity, and legality help programmers develop security devices and mechanisms based on the attack mode. The primary classification mentioned below is based on a survey conducted by researchers [4].

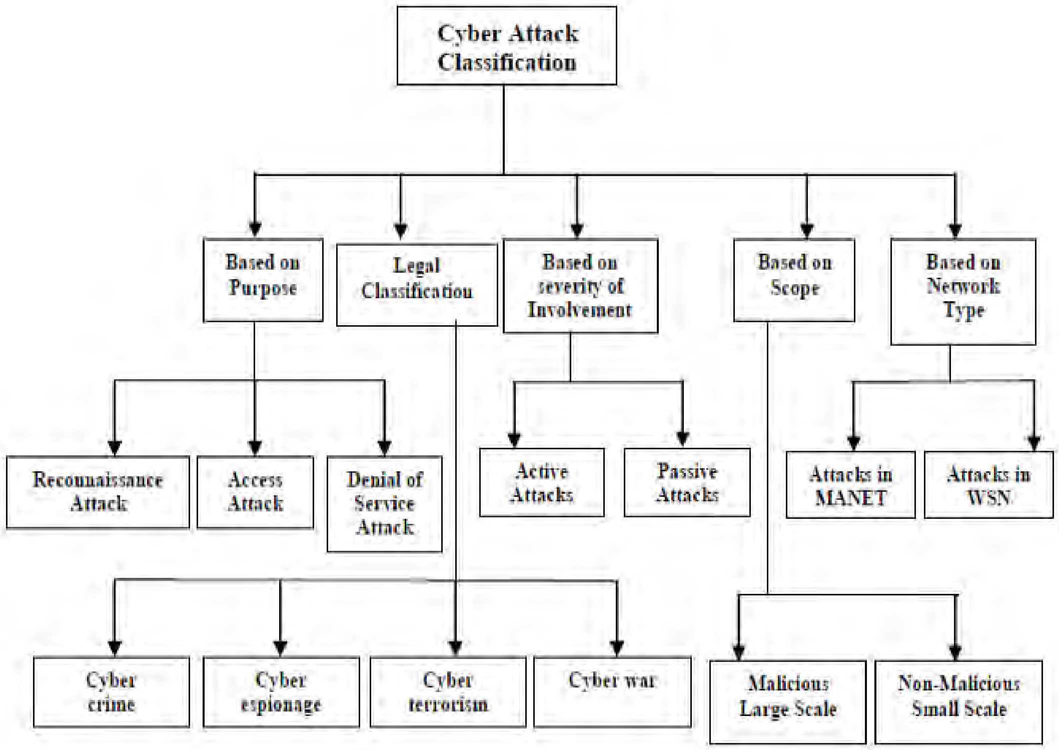


Fig. 1.2.Cyber Attack Classification Diagram [4].

Based on Fig. 1.2, cyber-attacks are classified into five categories based on purpose, scope, type of networks, severity of involvement, and legal classification. Usually, people want to disrupt networks for different purposes. They generally attack websites, organization databases, or target enterprise networks. Most attacks are generated using certain kinds of automated tools [9], which could have a massive impact on the networks. The authors include a few tools that hackers commonly use. These kinds of tools are readily available online. Attackers can freely download these tools and use them for malicious activities such as network mapping, trojan propagation, probe attacks, buffer overflow, application layer attacks, and DDoS attacks. Some tools are used in wired networks to capture and exploit valuable information, while others are used in wireless networks [1].

Similarly, various network security research groups and private security professionals have made many defense tools available in counterparts, including network information gathering tools like network mapping or sniffing tools and monitoring tools like visualization tools. In this research work, the focus was on Denial-of-service attacks. The data used here was collected from DARPA, and the network traffic file was affected by a DDoS attack on IP network nodes. Before moving further, the research will examine the DDoS attack background.

The most common cyber-attacks are DDoS attacks [19], which are known as Distributed Denial of Service attack. A denial of service is an explicit attempt by an attacker to prevent legitimate users from using network resources and overloading the network with illegitimate resource requests. An attacker attempts to "flood" a network and thus reduce a legitimate user's bandwidth, prevent access to a service, or disrupt service to a specific system or a user [10]. DDoS attacks were made by using two techniques, "Network-based attacks" and "Host-based attacks" [11], where the target could be a host, or it could be a whole network. People use DoS tools like Trinoo [12] to generate attacks. These tools are easily accessible to download from the internet.

Normal computer users can also become DoS attackers. DoS attacks in the network generally take down the target by exhausting its resources, which can be anything related to service performance or network computing, such as application/service buffers, CPU cycles, link bandwidth or TCP connection buffers. Individual attackers can also exploit the vulnerability, break into target servers, and bring down services. Since it is difficult for attackers to overload the target's resource from a single computer, many recent DoS attacks were launched via many distributed attacking hosts on the internet. These attacks were called Distributed Denial of Service (DDoS) attacks [11].

Aggregation of attacking traffic is tremendous compared to the available resources. It results in degraded network performance, or the network can collapse completely. Resulting DDoS attacks could be hazardous to the systems and relatively easy to manipulate the system performance. Unlike conventional DoS attacks, DDoS attacks are complex and harder to prevent with traditional firewalls and security systems, hence, many researchers are trying to take advantage of Data Science modeling to avoid such losses to organizations. Syn Flooding, ICMP Smurf Flooding [15], UDP Flooding [11], and Intermittent Flooding [16], are considered network-based attacks. In this research work, the DARPA DDoS attack dataset was used, which contained SYN Flood attack file. Let us look how the TCP SYN Flooding occurs.

TCP SYN flooding has a broad impact on many systems. As shown in Fig. 1.3 a client-server scenario, a client establishes a TCP connection to a server before any communication happens, also known as a TCP handshake. The client first sends an SYN message to the server, which the server acknowledges by sending an SYN-ACK message to the client. The client completes the connection establishment by responding with an ACK message. The server and the client connection are established now, and the service-specific data can be exchanged between them. The exploitation rises at the half-open state when the server waits for the ACK message from clients after sending the SYN-ACK message [17]. Instead of sending an ACK message, the client still sends multiple SYN messages to the server, which occupies all the resources, and there are no resources available for legitimate traffic.

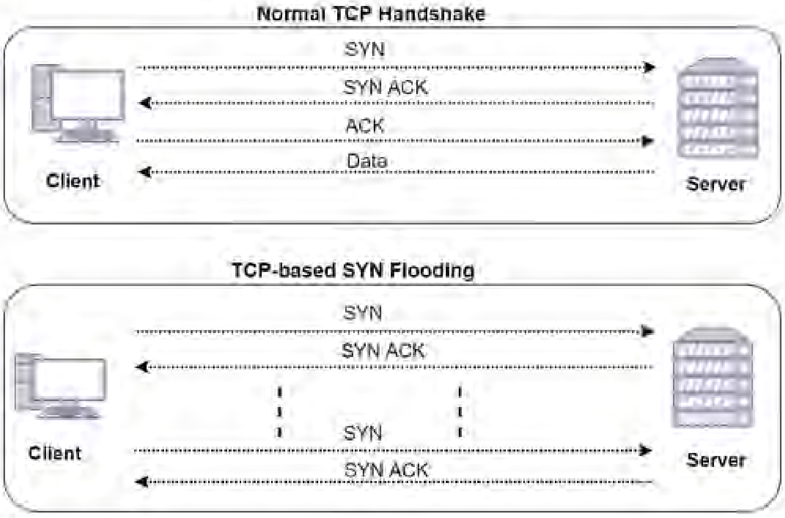


Fig. 1.3 Typical TCP Handshake (Top) and SYN Flooding Attach (Bottom) [18].

The server needs to allocate memory for storing the information of the half-open connection. The memory will not be released until either the server receives the final ACK message, or the half-open connection expires. Attacking hosts can easily create half-open connections via spoofing source IPs in SYN messages or ignoring SYN-ACKs. The consequence is that the final ACK message will never be sent to the target. Because the victim normally only allocates a limited size of space in its process table, too many half-open connections will soon fill the space. Even though the half-open connections will eventually expire due to the timeout, zombies can aggressively send spoofed TCP SYN packets requesting connections at a much higher rate than the expiration rate. Finally, the victim will be unable to accept any new incoming connection and thus cannot provide services. There are multiple ways to deal with such kinds of problems. Here, the research approach took advantage of Machine Learning and Deep Learning techniques to detect and predict the possible attack nodes.

Background of Deep Learning and Machine Learning Techniques

In the last couple of years, DL and ML technologies have been preferred by researchers across different disciplines. Similarly, DL and ML methodologies have demonstrated outstanding performance in the cyber-security field. Every sector in the digitized world is taking advantage of advancements ensued in the field of Artificial Intelligence. Extracting cyber-attack incident patterns or insights from network data and building respective data-driven models are key to making security systems intelligent and automated. Understanding, extracting, and analyzing valuable information from data using various scientific methods is commonly known as Data Science.

As the research focus narrows on technologies, Data Science for cyber-security is considered one of the significant applications in today's world [26]. The data is gathered from relevant cyber-security sources, and the analytics complement the latest data-driven patterns for providing more effective security solutions. This concept allows the computing process to be more actionable, automated, and intelligent than traditional ones in the cyber-security domain. There is an ample amount of cyber-security techniques that already exist in the market. Nevertheless, this work discusses and summarizes several associated research issues with the existing data science approach toward security and projects a naïve and sustainable solution that can be used on a small scale or even big scale data. Furthermore, I have provided ML and DL-based multi-layered frameworks for cyber-security modeling. Overall, the goal was not only to discuss cyber-security data science and appropriate methods but also to focus on the applicability of data-driven intelligent decision-making for protecting the systems from cyber-attacks.

Data science is broadly classified into three categories: Supervised Learning, Unsupervised Learning, Reinforcement Learning, and it takes advantage of techniques like Machine Learning, Deep Learning, Data Mining, and Artificial Intelligence. There is always confusion about the association between ML, DL, and AI. AI is a new technological science that studies and develops theories, methods, techniques, and applications that simulate, expand, and extend human intelligence [23]. Within AI, Machine Learning merged in not only the cyber-security field but also the choice for developing real-world software for natural language processing, computer vision, speech recognition, robot control, and other applications [24]. ML is like computational statistics and focuses on predictive modeling. It strongly relates to mathematical optimization, which delivers methods, theory, and application domains to the field. There is always a conflict between ML and data mining concepts [25], but the latter subfield focuses more on exploratory data-driven pattern analysis and is also known as Unsupervised Learning.

Deep Learning is a subset of Machine Learning. Its inspiration lies in establishing a neural network that simulates the human brain for analytical Learning. It mimics the human brain mechanism to interpret data such as images, sounds, and texts [27]. Deep Learning allows models composed of multiple processing layers to learn data representations with numerous levels of abstraction. Similarly, to ML methods, DL methods have Supervised and Unsupervised Learning methods. DL algorithms need a massive amount of data compared to ML algorithms to provide the correct predictions. The benefit of using DL is Unsupervised or Semi-supervised feature learning and hierarchical feature extraction efficiently replace features manually [28]. The Supervised Learning meaning dataset is already labeled, and the Unsupervised Learning dataset is not labeled.

Many studies have already been done in the field of Data Science Security, from modeling, tuning, training, testing, validation, and result analysis. The major focus of most of the studies was always on the prediction side of cyber-attack. This research work has given equal weightage to actions to take post detection of such events. If such events occur, even after preventive measures are taken, what could be the solutions to avoid the reoccurrence described utilizing this research? The solution lies in the concept called network monitoring. Further, in the next section, detailed information is provided.

Idea Behind Network Monitoring using the Visualization Tool

In generic terms, visualization means representing data in a certain way. It could be in terms of graphs, diagrams, flowcharts, or any mode of graphical representation to understand the data or information in an improved manner. Visualization of node data in the networking field is known as network monitoring. Monitoring networks is a key concept in network management as it helps network engineers determine a network's behavior and its components. Traffic engineering, quality of service, and anomaly detection also depend on monitoring for decision-making and organization performance management [29]. Visualization tools have emerged as an essential component, especially in medical, education, engineering, military, and environmental management. [30]. The enormous growth of networks where not only every household device is connected to a network but the tremendous amount of network setups in the industries led to the development of robust network monitoring or visualization tools. Cyber-space is also leveraging such powerful visualization tools to prevent cyber-attack incidents.

In this paper [31], the author has shown some studies on how cyber-attack monitoring systems, like geographical, temporal, and logical, are used. Each visualization had its advantages and disadvantages. Since it is crucial to analyze the information from different viewpoints and to make the right decision in practical cyber-attack monitoring, this visualization should be highly integrated. Every researcher has their way of using these visualization tools for the respected purpose of usage.

Similarly, this research study is also taking advantage of the visualization tool to reflect the network anomalies found by DL and ML models, which are nothing but DDoS attacks on the network. As well as representing comparison of predicted output with actual output in the graphical mode. so, the network engineer can take the decision or further actions accordingly. A detailed explanation of the visualization tool is given in the subsequent chapters.

Methodologies

Supervised Learning

Supervised Learning refers to training ML and DL models with labeled datasets in which algorithms train models to do prediction using data with known ground-truth output. Simply put, Supervised Learning algorithms learn by existing examples with labeled data. In the Supervised Learning approach, the dataset contains input data and their corresponding label [32]. The Supervised Machine Learning model learns through training using the most common algorithms like neural networks and Decision Tree. The classification algorithm is used for determining the error of the network and then the network minimizes the error to get reasonably extracted predictions. In Decision Tree, classifications are used to determine what attributes provide the most information that can be used to solve the classification problems. Optimization techniques are the key performers in the Supervised Learning models, which minimizes the error throughput.

When a model is trained using a Supervised Learning approach, the algorithm searches for unique data patterns that match the corresponding output. When training is complete with labeled data, a new set of unseen inputs are fetched to evaluate the newly trained model. The model will make predictions on the new inputs based on what is learned from the training data. The primary goal of a Supervised Learning model is to make the correct prediction of the label of new input data based on the existing labeled data. A Supervised Learning algorithm can be written as Equation 1.1

$$\mathbf{Y = f\,(x)} \qquad \mathbf{(1.1)}$$

Where x is the input variable and Y is the prediction. Supervised Machine Learning is broadly divided into classification and regression. These two classes are explained in detail in the next section.

Classification

The classification algorithm utilizes the data points with the corresponding category assigned to them. The classification algorithm's job is to take the input value and assign it to a corresponding class based on the training data. Fig. 1.4 demonstrates an example of binary classification. Here, a function generated by a binary classification algorithm classifies data into male and female classes. When a classification is carried out on only two classes, it is called binary classification. There are many classification algorithms, but this research focused only on Support Vector Machine (SVM), Decision Tree and Deep Neural Network (DNN). This research deals with a binary classification problem containing two classes viz "Normal" and "Anomaly" classes. Brief information on these algorithms is provided in the following section.

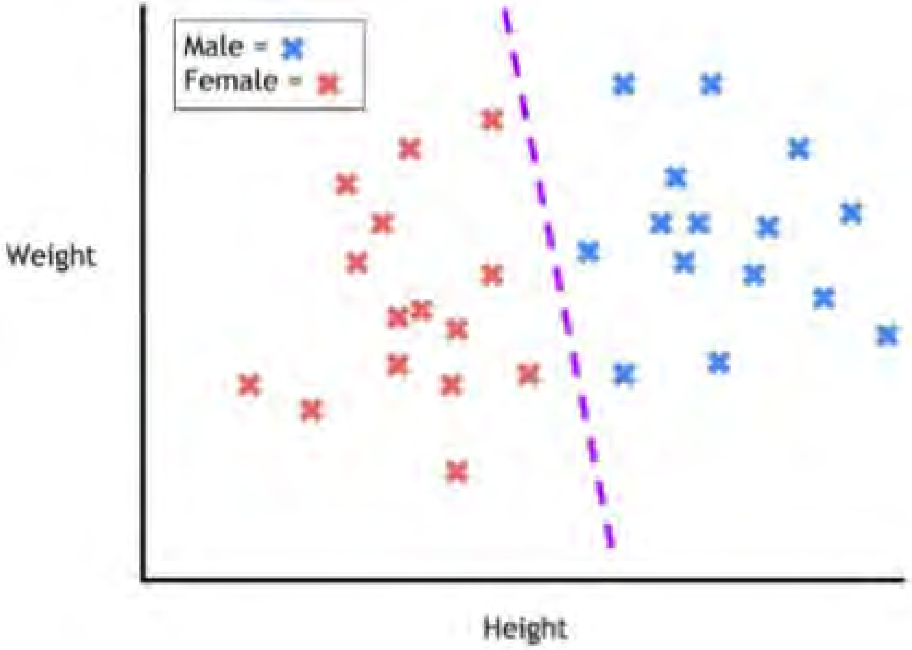


Fig. 1.4 Example of a classification algorithm [33].

Regression

In Machine Learning, "regression" is a statistical predictive process where the ML models find the relation between dependent and independent variables. The regression algorithm aims to make predictions of numbers such as revenue, temperature, and sales. Equation 1.2 shows a basic linear regression equation.

$$\mathbf{y\hat{} = w[0] * x[0] + w[1] * x[1] + ... + w[i] * x[i] + b \ (1.2)}$$

Equation 1.2 represents the features for the dataset, and w[i] and b are hyperparameters called weight and bias generated during training. Examples of regression algorithms are Linear Regression, Logistic Regression, etc. [33].

Support Vector Machine (SVM)

Support Vector Machine (SVMs) are Supervised Machine Learning algorithms that can be used for solving both classification and regression problems. However, SVM has been considered a significantly opt classifier to solve classification problems [34,35,36]. The accuracy of SVM classification is supposedly deemed high in predictive analysis, hence SVM models majorly used to resolve the classification problems. In the SVM algorithm, [38] each data item is plotted as a data point in N-dimensional space (N – number of features); then, it is the goal to find an optimal hyperplane that distinctly classifies the data points. There are many possible hyperplanes to separate the two classes of data, but the optimal hyperplane is the hyperplane that has a maximum margin from both the classes, as shown in Fig. 1.5.

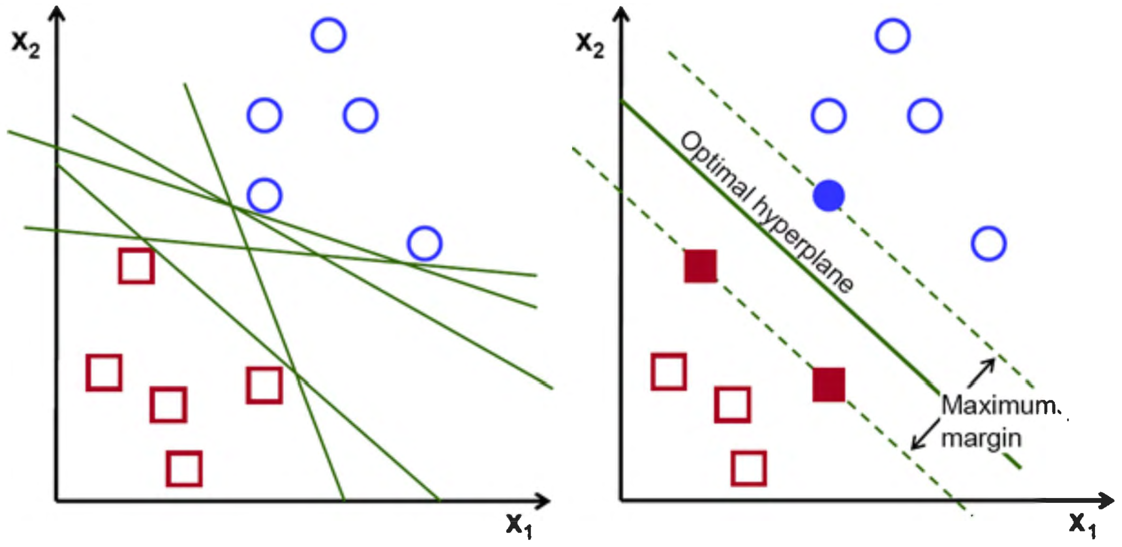


Fig. 1.5 Support Vector Machine (SVM) [38].

SVM works well on a small data set compared to vast and large-scale data [37]. The reason is the selection of hyperplanes that classify data points. Finding a suitable hyperplane to classify data correctly is difficult if the plane contains many data points. The classifier does not opt for large or complex datasets. The help of support vectors can influence the orientation of hyperplanes. Support vectors are the actual data points that are closer to the hyperplane. Using these support vectors, the margin can be maximized or changed as required. These are the points that help to build the SVM classifier.

Data points could have outliers, but SVM is designed to ignore the outliers from other classes. Hence, the SVM classifier is robust to outliers. It is easy to draw a linear hyperplane between the two classes, but a linear hyperplane cannot separate data points using a kernel technique [37]. The SVM kernel is a function that acquires low dimensional input space and does some extremely complex data transformations to transform it into high dimensional data space. The loss function Equation 1.3 that helps maximize the margin is the hinge loss function, explained well in [39].

$$\boldsymbol{c(x, y, f(x))} = \begin{cases} \mathbf{0}, & \textbf{if } \boldsymbol{y * f(x)} \geq \mathbf{1} \\ \mathbf{1} - \boldsymbol{y * f(x)}, & \textbf{else} \end{cases} \qquad \textbf{(1.3)}$$

This kernel concept in SVM is primarily valuable for non-linear classification problems. Cost function and gradient also play an important role in SVM.

Decision Tree

In Machine Learning, computer algorithms operate without being explicitly programmed. Various training and testing data determine the results in every condition. Decision Tree also is one of the techniques of Machine Learning, and it dramatically works with Supervised and Unsupervised Learning datasets. It uses three different algorithms called ID3, C4.5, and CART [40]. Decision Tree follows the same structure as Tree. It contains the root node, Decision node (branches), and leaf nodes [44] (see Figure 1.6). The root node is considered as the parent node and testing has happened on every internal node at attribute level. The determination of the test result for the branch and class label is dependent on the verdict of a leaf node. In simple language, a Decision Tree is a tree where each node represents a feature (attribute), each link (branch) means a decision (rule), and each leaf shows an outcome or a class (categorical or continuous value) [41,42].

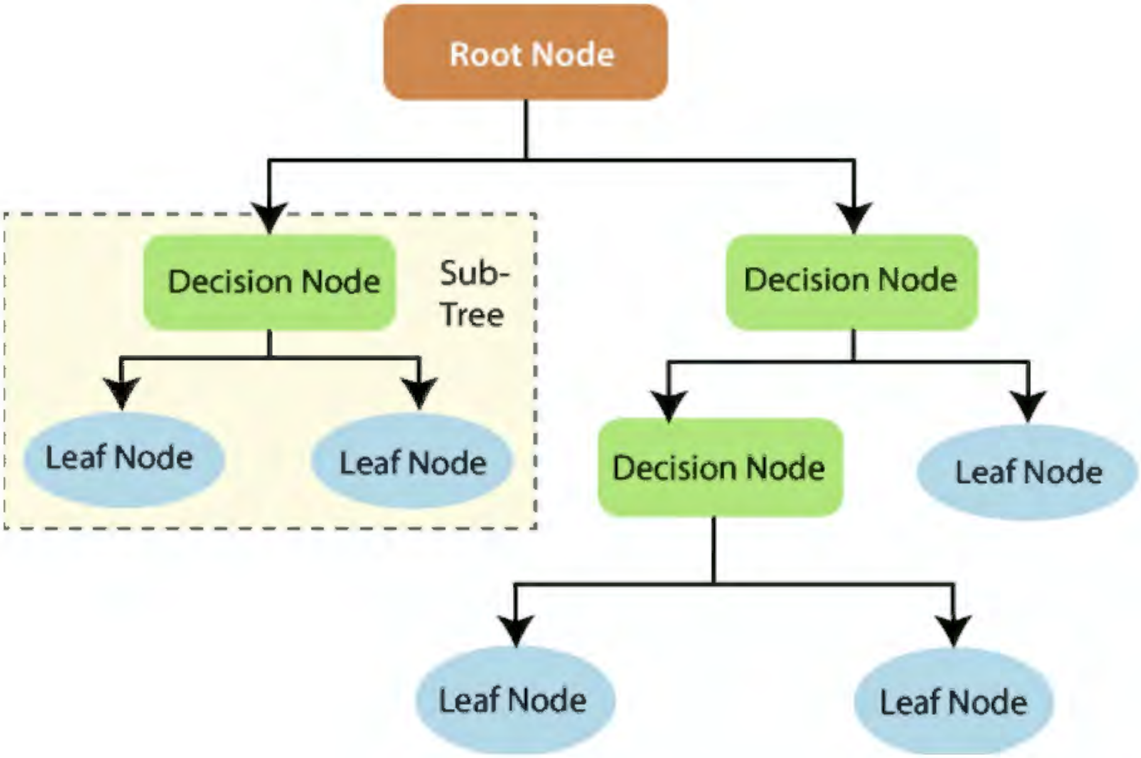


Fig.1.6 Example of Decision Tree [44].

Now the question comes, to start the splitting, which feature should be chosen as the root node [45,46]? Here, the feature's informativeness plays a role in selecting the root node. The more informative the feature, the feature gets selected to split the data on. This informativeness provided by the feature is called 'information gain.' It is highly related to the term entropy. Entropy is used to measure the impurity or randomness of the dataset. The value of entropy always lies between 0 and 1. Its value is worse when it is equal to 1 and better when it is equal to 0. The formula for calculating entropy is as follows Equation 1.4

$$\boldsymbol{Entropy(x) = -\sum(P(x=k) * \log_2(P\,(x=k))} \qquad \textbf{(1.4)}$$

the probability that a target feature takes is P(x=k) provided specific value, k. The feature with highest Information Gain is selected as root node to start splitting the Decision Tree The Information Gain formula is shown in Equation 1.5. ID3 uses the information gain.

$$\boldsymbol{InformationGain(feature) = Entropy(Dataset) - Entropy(feacture)}$$

(1.5)

The cart uses the Gini Index Equation 1.6. A feature with a lower Gini index is chosen for a split.

$$\boldsymbol{Gini\ Index = 1 - \sum(P(x = k))^2} \qquad \textbf{(1.6)}$$

The Decision Tree algorithm has advantages compared to other ML algorithms regarding noisy data. In this research work, Decision Tree has been selected as one of the algorithms because of how it handles the missing values and its data manipulation techniques. A standard – but incorrect method of handling missing data is to exclude the missing values cases, which increases the risk of introducing bias in the analytical result of prediction and hence inefficient technique. It deals with missing values in two ways. One, it classifies missing values as a separate category that can be analyzed with other classes. The second is making missing values as target variables, then making predictions, and finally replacing these missing ones with predicated results [43]. The only drawback with the DT algorithm is that its calculation complexity may increase with more training samples. It is incapable of doing justice to the large datasets. Hence the third algorithm selected for this research work is the Deep Neural Network (DNN). It is capable of handling large datasets.

Deep Neural Network (DNN)

A Deep Neural Network is nothing but an Artificial Neural Network with several hidden layers, usually more than five hidden layers within. DNN, as well as ANN, belongs to Deep Learning Mechanism. DL is a field within ML and AI that deals with algorithms inspired by the human brain to succor machines with intelligence without being explicitly programmed [47,48]. ML worked well for various simple classification and regression problems but failed to outrival mostly with issues like images, audio, and other unstructured data types. Most researchers parallelly work on Neural Networks, which mimic the human brain's biological process, which comprises billions of neurons. However, these NNs made limited progress due to limited data and computation power as well as the complexity of data. Then the DL field led to solving the shortcomings of ML and NN. DL has improvised Neural Networks with many more hidden layers to act like brain neurons to process the complex data. Machine Learning mainly performs poorly with image, audio, and other unstructured data types. While DNNs have become famous for their extraordinary success in various ML and DL tasks, see for more details (49,50,51,52).

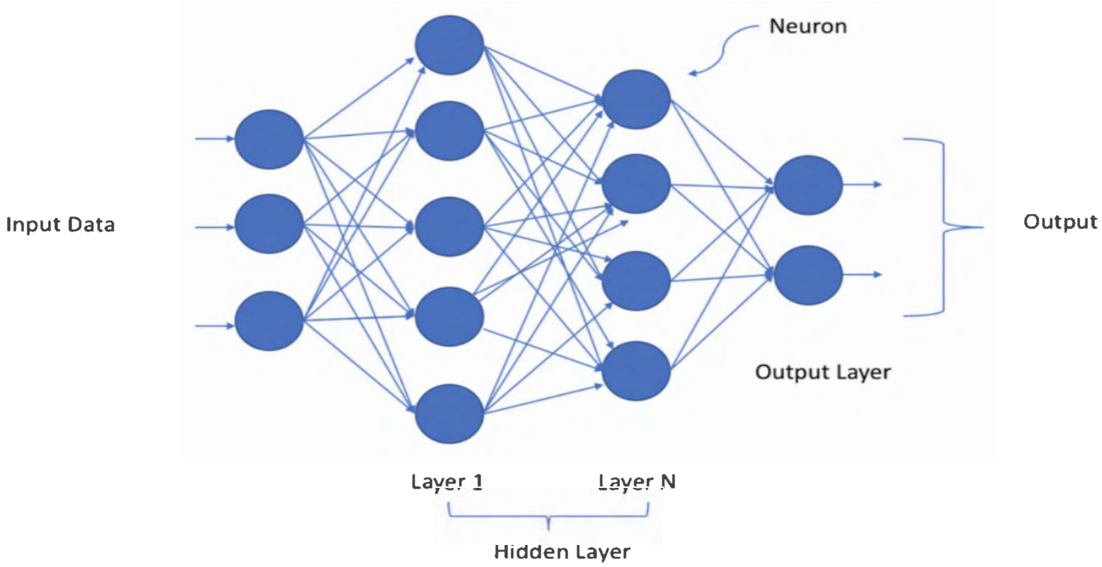


Fig.1.7 Deep Neural Network with N hidden Layers [47].

A simple DNN network is shown in Fig. 1.7. It is a hierarchical structure of neurons, including an input layer and an output layer with N number of neurons hidden in layers. These neurons are connected and pass a signal to other neurons based on input data and learn by using some feedback mechanism. Every neuron computes a small function, i.e., the Activation Function. Refer to Fig. 1.8 for block diagram of Activation Function. Neuron layers would have associated weight and bias. This weight defines the influence of the input on the output layers. Initially, the initial weights would be random, however, neurons update the weights iteratively during the learning process to minimize the loss function and cost function [54].

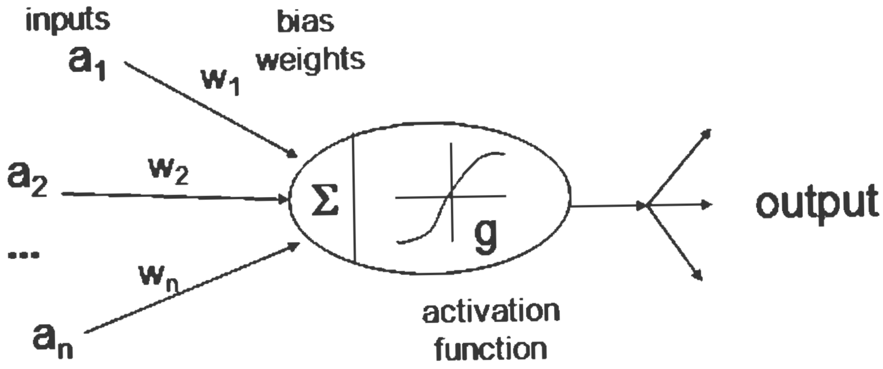


Fig. 1.8 Block Diagram of Activation Function [54].

The output of DNN is in the form of predictions like Yes or No, or it could be a probabilistic value, and it depends on the activation function used in output layer neurons to get correct predictions. Activation Function are divided into two categories: linear Activation Function and non-linear Activation Function, i.e., Sigmoid, Tanh, ReLu etc. [53]. Here the Sigmoid Activation Function has been used, represented in Equation 1.7. Sigmoid has been selected to get output in binary classification form, and the output we wanted is either 1 or 0.

$$\phi(z) = \frac{1}{1+ e^{-z}} \qquad \textbf{(1.7)}$$

Visualization Tool for Network Monitoring

In most of the ML and DL research related to Cyber-security Intrusion Detection problems, performance analysis is analyzed by dint of evaluation matrix, that is, accuracy, precision, recall and F-score, and Confusion matrix [55,56,57]. Along with the evaluation matrix, this study opted for a visualization tool as a performance analysis practice. After extracting useful information from a large dataset using different ML and DL models, prediction results were displayed in an easy-to-interpret visualization format. This visualization tool takes advantage of the data representation technique for effortless network monitoring for attacked nodes in the form of a graphical representation of IP nodes. Finally, the tool compares existing and predicated labels for attacked nodes on the visualization screen and indicates nodes with different color codes. The methodologies used to build the frontend comprise IIS, Cytoscape tool open platform tool libraries, HTML, CSS, and JavaScript, as explained in Chapter 3. Fig. 1.9 shows the look of visualization tool.

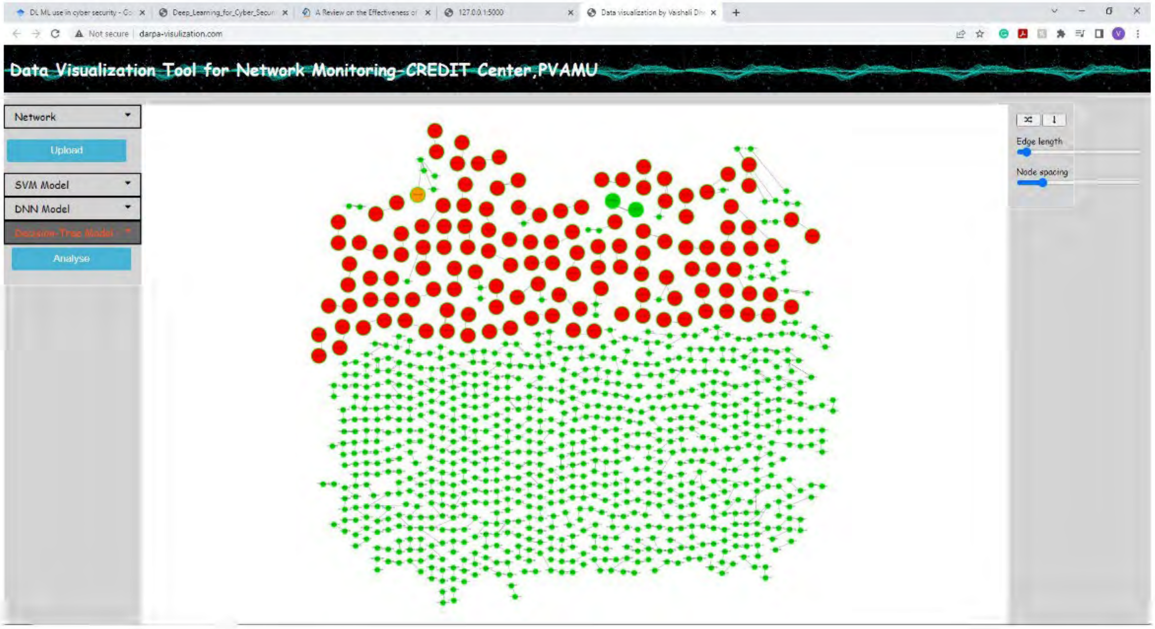

DL ML use in cyber security - Go
Deep_Learning_for_Cyber_Secur
A Review on the Effectiveness of
127.0.0.1:5000
Data visualization by Vaishali Dh
Not secure
darpa-visulization.com
Data Visualization Tool for Network Monitoring-CREDIT Center,PVAMU
Network
Upload
SVM Model
DNN Model
Decision Tree Model
Analyse
Edge length
Node spacing

Fig. 1.9 Visualization Tool for Network Monitoring.

The detailed description and methodology used is explained in subsequent sections.

Mission Statement

Scope

The tremendous growth of IP networks with this digital revolution led to the explosive growth of cyber-attacks on networks. This unethical practice is exploiting the organizations until they collapse entirely. Hence, appropriate defense measures are necessary to meet the industry demand. ML and DL-based model solutions to predict the possible attack using an existing malicious traffic pattern depend on proper model selection at the backend. This study revolved around selecting multiple ML and DL algorithms to handle diversified and scalable data. IP Network Monitoring plays an indispensable role in cyber-attack scenarios. A visual representation of targeted IP nodes in the giant networks is crucial to identify in terms of taking defense measures promptly. Therefore, the need to correctly integrate ML-DL predictive modeling at the backend and visualization tool for network monitoring at the frontend were a challenge with this study.

In the middle of this cyber-crime crisis, developing the DL, and ML models, building the visualization tool, and integrating them for network monitoring has necessitated this research work.

Contributions

The main goal of this work was to build a Deep Learning enhanced visualization framework for network monitoring to take defensive measures against increasing cyber-crime. This study proposed a loosely coupled, flexible intrusion detection and network monitoring framework. The frontend (visualization tool) or backend (DL models) can be replaced easily with any other alternatives as needed.

The goal was to build a framework with minimal financial expenditure, where every small-scale industry can also leverage this framework, who could not afford expensive intrusion detection systems. In addition, this study aimed to extend the work on Deep Supervised Learning models for network monitoring for the first time using open platform visualization tool libraries to achieve better performance.

Challenges

Many challenges have come along with the exhilaration of new inventions. These challenges can be categorized into three broad categories stated below.

Data Related Problems

Data is one of the key factors in performing any project in Data Science. Deep Learning is a data-intensive process. A lot of data is needed to train the models to perform at high accuracy. The learning process of the models intensively depends upon the amount of data available, and the model's predictive power relies on the quality of data used. Similarly, the dataset used for Supervised Learning needs to be accurately labeled, which requires a lot of manual work. Labeling these massive datasets is an expensive as well as time-consuming process. Hence the availability of a gigantic, labeled dataset with good quality from reliable sources was the challenge. The dataset used in this work had a field like a source IP address and a destination IP address. Models so far cannot directly process special characteristics like "." as IP addresses, that is, 10.10.1.120. Data modification was the obstacle to handle.

Difficulty in Model Selection

The selection of models in DL is a crucial process. If the wrong models are selected, it could cost time and money. Model selection was the essential and challenging part of the work performed. Suppose the plan is to use the same size dataset for all three models. In that case, model performance is not satisfactory because each model requires different data sizes to perform to the standard. For this study, the plan was to use the different chunks of the same dataset from small to large scale so the project could be scalable.

- The Decision Tree model was chosen because it deals with outliers and can work with small to moderate dataset sizes.
- SVM was the second choice of model since it works great on small datasets, and the performance is up to the mark.
- DNN model is the excellent third choice to deal with massive datasets. In most Deep Learning predictive modeling, Neural Network is the best-suited option to find the patterns and deep feature extractions from complex datasets.

Visualization Tool Related Issues

Many successful visualization tools are already present in the industry. Tableau, Orange, and Power BI are a few popular ones. But restrictions with these tools are, first, pre-owned and cannot be used free of cost. Second, it cannot be integrated with another backend system as needed, and the last one is the tool's scalability to represent the number of IP nodes in the network. So, a tool was needed that could be modified according to project requirements and could be cost-effective, so anyone could take leverage of technology without spending a single penny. Hence, an open platform visualization tool called Cytoscape JavaScript-based visualization library was selected to overcome issues with pre-existing visualization tools. Cytoscape is a scalable tool, but one disadvantage is that it needs a high-power graphic card to represent the vast data nodes, which is a little setback in the project. All information about the tool is given in subsequent sections.

Data Frontend and Backend Integration Problems

The research work scope also included the integration part of the Backend (DL models) with the frontend (Visualization Tool). The researcher intended to project the visualization tool on a web page, meaning hosting the visualization tool as a website on the

HTML page. Cytoscape libraries had limitations on the format of data received from the backend to the frontend. It was programmed to receive data only in JSON format. So, at the backend in the data science model, it was necessary to convert the data into JSON format before passing it to the visualization tool. This integration was done with the help of API (Application Programming Interface). Selection between REST API and DJANGO API was a tough call to take. Selection of REST API was made over Django, keeping all conditions and requirements in mind. More information is provided in the following sections.

Outline of the Study

The study consisted of the following sections,

- Chapter 2 contains Literature Review.
- Chapter 3 represents the methodology used in the research work. It is divided into three significant parts. First, ML and DL modeling for predicting cyber-attack including data preparation and preprocessing and statistical analysis to get desired output. Second, it focuses on methodologies used in designing a Visualization tool, and finally, backend integration for network monitoring.
- Finally, Chapter 4 enumerates the details of the datasets used for the implementation, experimental setup, evaluation methods, and performance analysis of the three models presented in this study.
- Chapter 5 concludes the study and highlights potential future work and research.

II. LITERATURE REVIEW

Literature Review

The complexity and severity of cyber-attack on networks is increasing over the period. Studies show the demand for defense mechanisms to protect the network from cyber threats [1,5,6,7]. Many surveys have been conducted on cyber-attack, their classification, and attack detection strategies [2,4]. Nowadays, cyber-attacks are conducted with the help of cyber-attack tools, so the impact on the systems could be massive and cause utmost harm to the companies [1,9]. As a result, counterpart defense systems should be diversified and capable of dealing with the severity of attacks generated by attackers. The most common attack type is a DDoS attack. A lot of review papers on DDoS attack at an aggregated level already exist [17,19]. Some of them contain the description and the purpose [10,11,13], economic impact [8], and few methodologies (tools) used for attacks [9,12]. This section reviews the literature on cyber-attack prevention practices attempted by researchers, which includes the traditional approach. Furthermore, this chapter dives deep into various supervised ML and DL methods used for predictive modeling to evaluate performance. Also presented are the work references visualization tools related to cyberspace protection technology.

Traditional Approach for Cyber-attack Detection

In a traditional approach, the main goal of a network defender is to reduce abnormal activities from live network traffic by monitoring the activities on a regular basis. This goal is somewhat achieved with the help of certain network security tools. Example tools include Orebaugh et al.'s [60] the "Wireshark" Analyzer tool kit. "Ntop" is another real-time network monitoring tool [61] by Deri et al. Also, LOIC [62] by Pras et al., the original

“LOIC” Tool, was built by Praetox Technologies as a stress testing application and then used to detect simple DoS attacks. Similarly, “HOIC” Mansfield-Devine et al. [63] is also a network monitoring tool. These types of tools are predominantly used for capturing live network traffic. These security tools are built in such a way that they perform activities like preprocessing, and feature extraction and analysis. [1]. Mansfield-Devine did justified work with actual attack detection as well as visualization. Nonetheless, these tools have their own limitations and failed to detect intrusion with the current scenario of big networks.

Learning-Based Approach

Statistical Approach

In a statistical approach, usually predetermined distributions of normal and abnormal behaviors of network traffic are applied to the models. In the defense scheme of [72], the authors applied a statistical approach where distance measuring schemes were used in addition to the ML classifiers like K-means, SVM, Decision Tree , and Naïve Bayes algorithm. Furthermore, a traditional technique is included in this study [73] called D-WARD. It is a statistics-based DDoS defense scheme. This paper [74] presents methods to identify DDoS attacks by computation of entropy and frequency-sorted distributions and explains detection algorithms, e.g., Entropy and Chi-square statistics.

Anomaly Detection Approach by Machine Learning and Deep Learning

DDoS attack survey and comparison presented in the paper were carried out by various authors and depicted the need for defense mechanisms against attacks stated by Bhardwaj and team [68]. Various ML and DL techniques have been studied by researchers and used in the past successfully to protect against DDoS attacks [19],[20],[22],[26].

Intrusion detection is classified into two types: anomaly and misuse detection [65]. An anomaly detection system creates two different datasets; one is a normal behavior dataset, and the second is a deviation from normal behavior. When deviation occurs, it triggers the occurrence of intrusion. The misuse detection system stores a predefined attack pattern in the system, and when a similar pattern occurs, it is classified as an attack. Studies indicate that Support Vector Machine (SVM) stands out as the top choice for tackling classification problems. However, a drawback of the SVM model is the extended training time it requires when dealing with sizable datasets. Several security researchers have worked with SVM-based IDS Intrusion Detection System (IDS) long ago [64][65][66]. They tried to overcome the processing training speed of SVM by taking multiple approaches. To help classify cyber-attacks, Saxena and Richaariya [64] proposed an intrusion detection system based on data mining techniques such as SVM along with particle swarm optimization methods for attaining a higher detection rate on KDD Cup 99 dataset. To overcome the training time and speed up the process of SVM, author [66] applied reduction techniques using clustering analysis to approximate support vectors on the 1998 DARPA dataset.

In this research paper [67], a Genetic Algorithm (GA) was proposed to improve Support Vector Machine (SVM) based on IDS. In a study conducted by scholars [69], a combination of the Naive Bayes technique and K-means clustering was employed to detect DDoS attacks. The approach involves a dual process: (1) Utilizing the Naive Bayes algorithm for categorizing labeled data clusters, and (2) grouping similar data based on their behaviors through clustering, followed by labeling all data based on K clusters. Data mining schemes like Naïve Bayes and clustering schemes mainly focus on Unsupervised Learning than Supervised Learning.

Modeling with ML techniques has its limitations, thus, there arises the need for a better model to tackle this problem with complex and giant datasets as none of the classical methods could perform as expected. Thus, Deep Learning methods were employed. Saied A, Overill RE, Radzik T (2016) and their colleagues [70] harnessed Neural Networks to identify both familiar and unfamiliar DDoS attacks. They achieved a remarkable accuracy of 98% using Artificial Neural Networks (ANN). The study shown in [71] took advantage of the computer vision technique where the traffic records were considered as images and detecting the attacks was viewed as a computer version issue. Although informative of the state-of-the-art in intelligent DDoS detection and prevention, the intricacies and inherent technical implementation details of the proposed schemes stand in the way of a fair comparative assessment of the studied schemes, thereby affecting the outcome of the analysis.

The Deep Learning concept was proposed by Hinton [76] based on the deep belief network (DBN), in which a layer-by-layer training algorithm was proposed to help the optimization problem of a deeply layered structure. Then the deeply layered structure of a multi-layer automatic encoder was proposed. In addition, the convolution Neural Network proposed by LeCun et al. [77] reduces the number of parameters to improve the training performance. DNN has been used in extraordinary work successfully in a variety of Machine Learning tasks, like Image Classification, Traffic Sign Classification, ImageNet classification, and acoustic modeling in speech recognition (49,50,51,52). Now this research work takes advantage of solving classification problems in intrusion detection cyber-security systems. Paper [75] has provided numerous ML, and DL algorithms with their challenges which mention DL is not fully accepted by the industry because the results provided by the models are hard to explain. As much as the prediction of attacks is important similarly, network monitoring with the help of visualization is also equally crucial. A few of the methods are explained in the following section.

Visualization Technique

"STARMINE," named visualization system for cyber threat monitoring, is presented in [78] and describes three different views of the cyber threat as, logical view, geographical view, and temporal view, in 3-D space. Similarly, many big companies have their own intrusion detection systems designed and established, which are not affordable for small-scale industry owners. There are a lot of open platform visualization tools available in the market which can be modified according to the need and then are implemented for monitoring the network. Cytoscape is one of the open-source library tools used in this work and is integrated with backend ML and DL models to show the attacked nodes on the network.

III. METHODOLOGY AND EXPERIMENTAL SETUP

Overview

This chapter focuses on dataset details, data preparations performed on the dataset, and methodologies used for the backend and frontend of the project. This work showcases the ML and DL modeling and implementation setup details with a flowchart. It then elucidates the approaches taken for building the frontend user interface for the visualization tool and, in the end, explains the technique used for integrating the frontend and backend.

With comprehensive elaboration provided in the preceding section, let's now delve into an overview of the complete setup and methodologies in brief. Fig. 3.1 shows the overall ideology behind this research work and the connection between models and visualization tools.

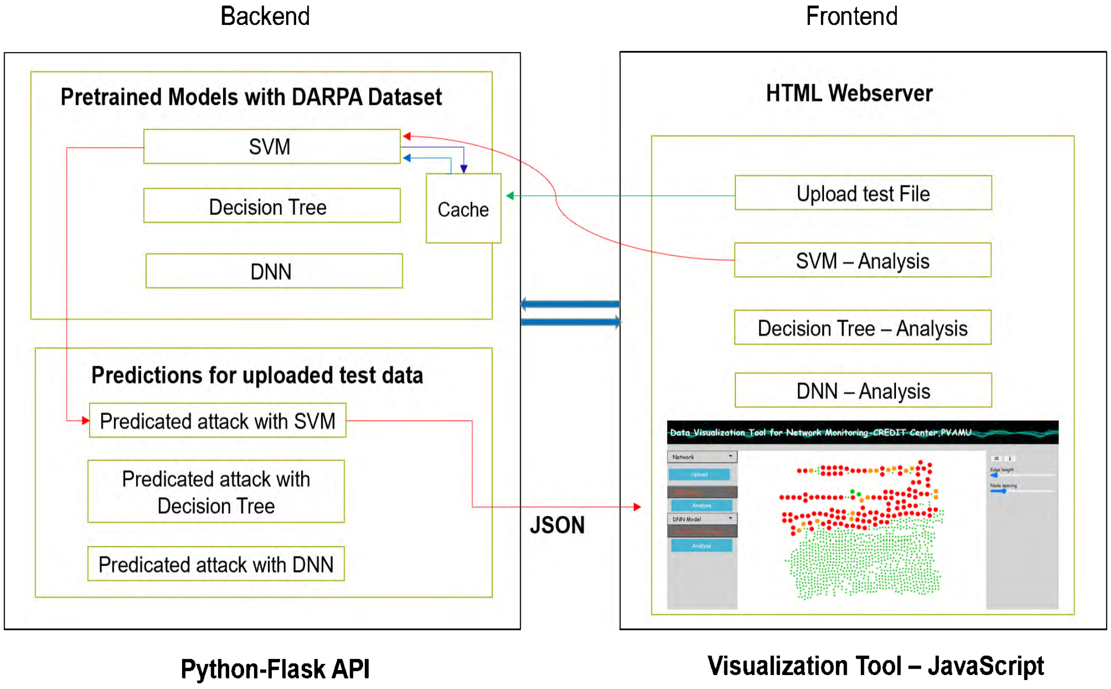

Backend
Frontend
Pretrained Models with DARPA Dataset
SVM
Decision Tree
DNN
Cache
Predictions for uploaded test data
Predicated attack with SVM
Predicated attack with Decision Tree
Predicated attack with DNN
JSON
HTML Webserver
Upload test File
SVM – Analysis
Decision Tree – Analysis
DNN – Analysis
Data Visualization Tool for Network Monitoring-CREDIT Center,PVAMU
Network
Upload
SVM Model
Analyse
DNN Model
Decision Tree Model
Analyse
Edge length
Node spacing
Python-Flask API
Visualization Tool – JavaScript

Fig. 3.1 Overview of Methodology.

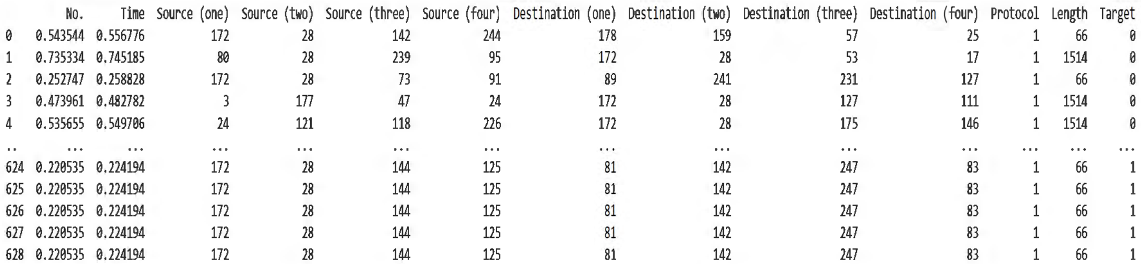
The backend consists of three Data Science models: SVM, Decision Tree, and DNN. All data pre-processing, feature engineering, and data preparation work are done at the backend. Python programming language has been used to write the program for modeling, and PyCharm IDE is the environment elected to write the code. Cytoscape is a visualization tool that is majorly used in the medical field to demonstrate protein synthesis or genetic molecular representation, and the reason is the scalability tool offers. Here the Cytoscape tool is modified according to the requirement and used for building the frontend. It is an advantage of the open-source visualization tool library taken to the best of its abilities to show IP nodes and traffic in-between. The tool is written in the JavaScript programming language. Likewise, HTML protocol is used to host the tool on the web browser that is nothing but a website "www.darpa-visualization.com." The style format, color coding, and for designing of webpage CSS coding has opted. The visual studio framework is used to write the code for HTML and CSS. Finally, Flask-API technology connects the ML DL backend to the visualization tool.

All these mythologies are explained in upcoming section 3.5. Firstly, the following section describes the dataset details along with pre-processing techniques considered.

Dataset Details

Data Set

The DARPA DDoS attack dataset has been used in this study to perform modeling. "DARPA_2009_DDoS_attack-20091105" dataset is provided by the LANDER project and described as an SYN flood DDoS attack metadata. This dataset is part of the 2009 DARPA Scalable Network Monitoring (SNM) Program Traffic. Dataset is a labeled dataset with a target provided. It is a classification problem-based dataset with two classes: Target IP address under attack (class 1) and Target IP address not under attack (class 0). It contains a synthetically generated traffic file for six minutes consisting of SYN flood attack records and legitimate IP traffic. These are PCAP files with fields like source Ip address and destination Ip address, timestamp, protocol length, and target to minimize the pre-processing time in this research work, a dataset with some statistical operations performed by Brandon Williams et al. [58] was used to proceed with further work. The pre-processed dataset is shown in Fig. 3.2.



```
          No.     Time Source (one) Source (two) Source (three) Source (four) Destination (one) Destination (two) Destination (three) Destination (four) Protocol Length Target
0    0.543544 0.556776          172           28            142           244               178               159                  57                 25        1     66      0
1    0.735334 0.745185           80           28            239            95               172                28                  53                 17        1   1514      0
2    0.252747 0.258828          172           28             73            91                89               241                 231                127        1     66      0
3    0.473961 0.482782            3          177             47            24               172                28                 127                111        1   1514      0
4    0.535655 0.549706           24          121            118           226               172                28                 175                146        1   1514      0
..        ...      ...          ...          ...            ...           ...               ...               ...                 ...                ...      ...    ...    ...
624  0.220535 0.224194          172           28            144           125                81               142                 247                 83        1     66      1
625  0.220535 0.224194          172           28            144           125                81               142                 247                 83        1     66      1
626  0.220535 0.224194          172           28            144           125                81               142                 247                 83        1     66      1
627  0.220535 0.224194          172           28            144           125                81               142                 247                 83        1     66      1
628  0.220535 0.224194          172           28            144           125                81               142                 247                 83        1     66      1
```

Fig. 3.2 Pre-processed data obtained from reference paper [58].

Python programming cannot perform operations on direct IP addresses (mainly on special characters like "."). Henceforth, the IP address was split into network and host addresses, as shown in Fig. 3.2. Source Address is divided into four columns e.g., Source(one), Source(two), Source(three) & Source(four), likewise the destination IP address. The rest of the fields were kept as they were.

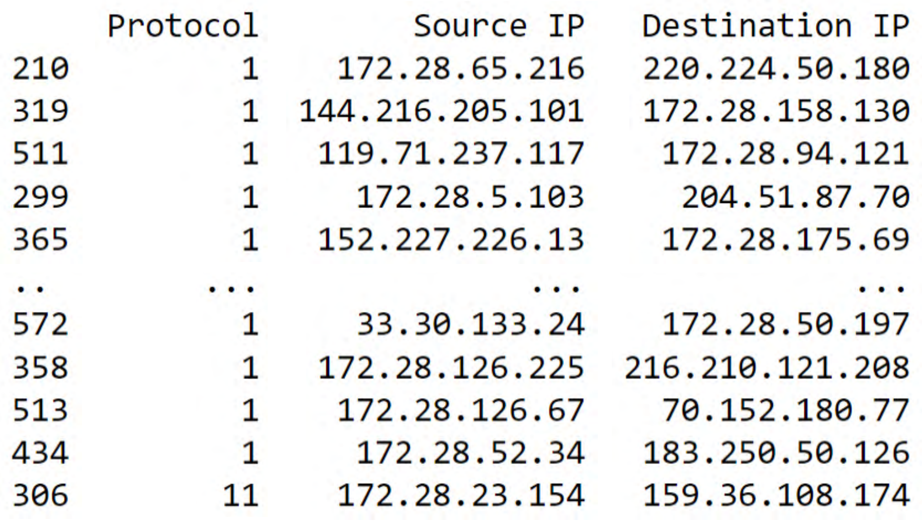
Data Pre-processing

Pre-processing is called cleaning and transforming raw data into a more usable and required form. In every work, pre-processing of data takes maximum time and effort during the data science modeling process. Python programming language has been used to do the all-pre-processing work in this research. Data cleaning techniques like missing values, zero values, and data distribution were checked, but as this dataset was already pre-processed, it did not have many outliers or errors to clean. Data Normalization was not done because it distorts the IP addresses (subnet and host address) required as it is in the dataset. Proceeding further modeling code was executed for three models for training and testing purposes. By using evaluation matrixes, output was evaluated.

Feature Engineering

Core Feature Engineering for ML and DL

Feature Engineering is a process of selecting, transforming, and manipulating raw data into features. Here chi-square feature selection technique was used in the Decision Tree algorithm. In this data set, attributes called "Number," "Time," and "Length" were not closely related to the requirement, so they were removed from the dataset. In the process of new data preparation, source and destination IP addresses were needed. Henceforth, attributes Source(one), Source(two), Source(three) and Source(four) were transformed into a single attribute as “Source IP address.” Similarly, attributes Destination(one), Destination(two), Destination(three) and Destination(four) were transformed into “Destination IP address” as shown in Fig. 3.3



```
     Protocol       Source IP   Destination IP
210         1    172.28.65.216   220.224.50.180
319         1  144.216.205.101   172.28.158.130
511         1   119.71.237.117    172.28.94.121
299         1     172.28.5.103     204.51.87.70
365         1   152.227.226.13    172.28.175.69
..        ...              ...              ...
572         1     33.30.133.24    172.28.50.197
358         1   172.28.126.225  216.210.121.208
513         1    172.28.126.67    70.152.180.77
434         1     172.28.52.34   183.250.50.126
306        11    172.28.23.154   159.36.108.174
```

Fig. 3.3 Pre-processed dataset overview.

Feature Engineering for Visualization Tool

The visualization tool used in this research is called Cytoscape, a JavaScript-based open platform tool library. This tool library was modified and used per the project's requirements. Many parameters have been introduced to generate backend data understandable by the frontend. For example: 'Dest_IP_count, '1s_dest_Count' and 'count' were used to calculate the "Weight and Score" parameters to show edge thickness between two IP nodes and the color of the IP nodes. Hence, two main parameters were acquainted with the backend code. The first parameter is called "Weight" and "the second parameter is called "Score," respectively. The final dataset after feature engineering is shown in Fig. 3.4

Added Attributes:

1. Dest_IP_count: Represents the Destination IP address instance that has occurred for how many times in the dataset with respective Source IP address.

2. 1's_dest_count: Represents the respective count of Destination IP address with Target attribute 1 or class "1" has occurred in the dataset with the respective source IP address.

3. Count: Represents the count of Destination IP addresses that has occurred in the dataset irrespective of the Source IP address.

4. Weight: Represents the thickness of the line and color between the two nodes and is calculated as a group by function as follows with respect to destination IP count. Conditions for the Edge color and thickness between two nodes are shown in Fig. 3.4

Weight = df.groupby (["Destination IP","Source IP"])["Destination IP"].transform("count")

a. Red color 🟥 link indicates: traffic between nodes is abnormally large-scale traffic, and if it is connected to the red color node, it is possibly the DDoS attack traffic.

b. Orange color 🟧 link indicates: traffic between nodes is mid-scale traffic, and if it is connected to the red color node, it has moderate chances of DDoS attack traffic.

c. Green color 🟩 link indicates: traffic between nodes is regular traffic.

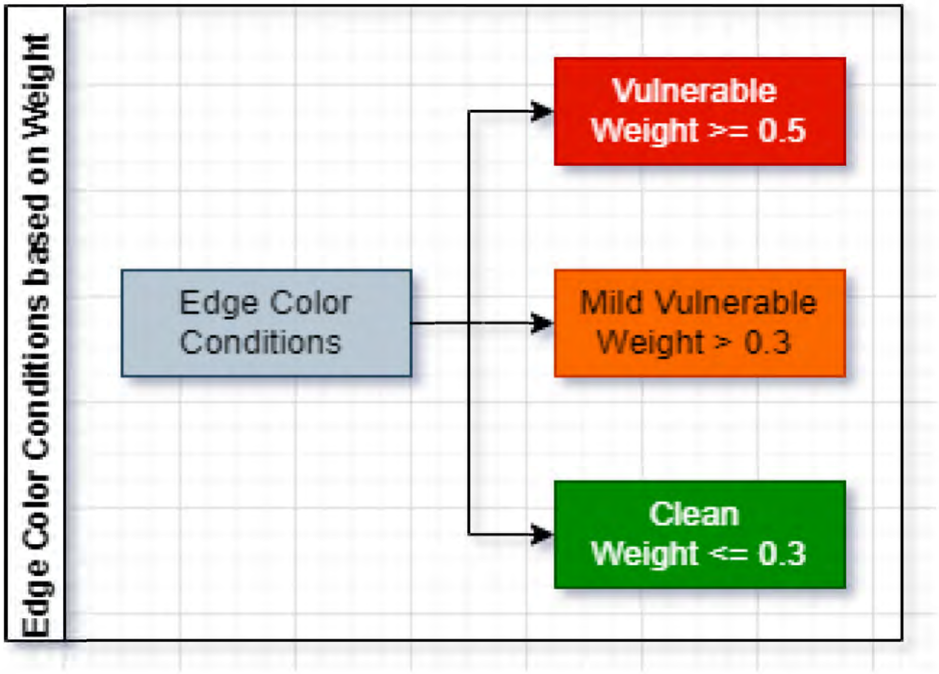

Edge Color Conditions based on Weight
Edge Color Conditions
Vulnerable Weight >= 0.5
Mild Vulnerable Weight > 0.3
Clean Weight <= 0.3

Fig. 3.4 Edge Color Conditions.

Score: represents the color of the node's color depending upon the predictive analysis of whether the node is under attack, moderately under attack, or normal node. Conditions for the node color is shown in Fig. 3.5. A formula for score is shown in Equation 3.1

$$Score = \frac{1s\_Dest\_Count}{Dest\_IP\_Count} \qquad \textbf{(3.1)}$$

a. Red color 🟥 node indicates: the node is under attack by source IP node.

b. Orange color 🟧 node indicates: the node has moderate chances of being attacked by the source IP node.

c. Green color 🟩 node indicates: the node not under attack by the source IP node.

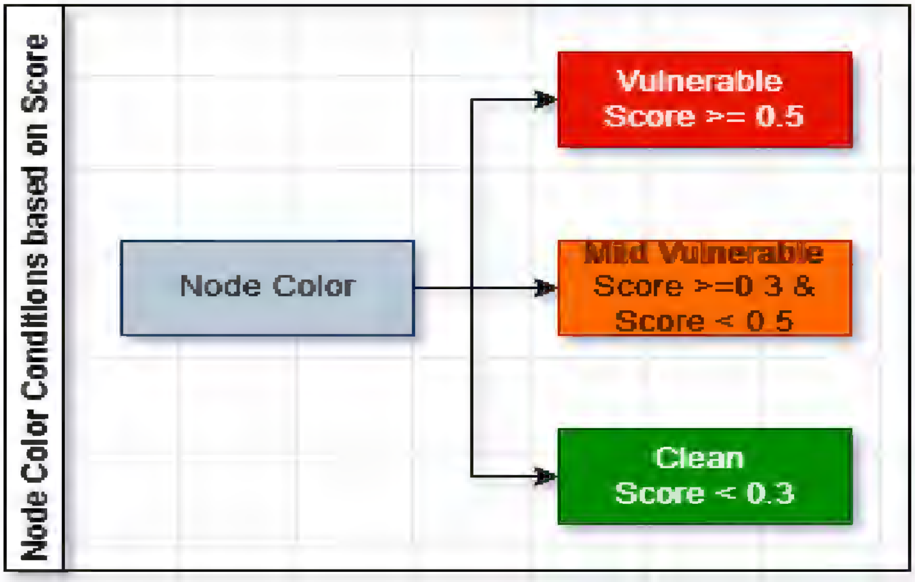


Fig. 3.5 Node Color Conditions.

Fig. 3.6 below shows the visualization tool GUI representation. All the details related to the front-end tool are explained in the following sections. Final dataset view after all preprocessing and feature engineering is shown in Fig. 3.7.

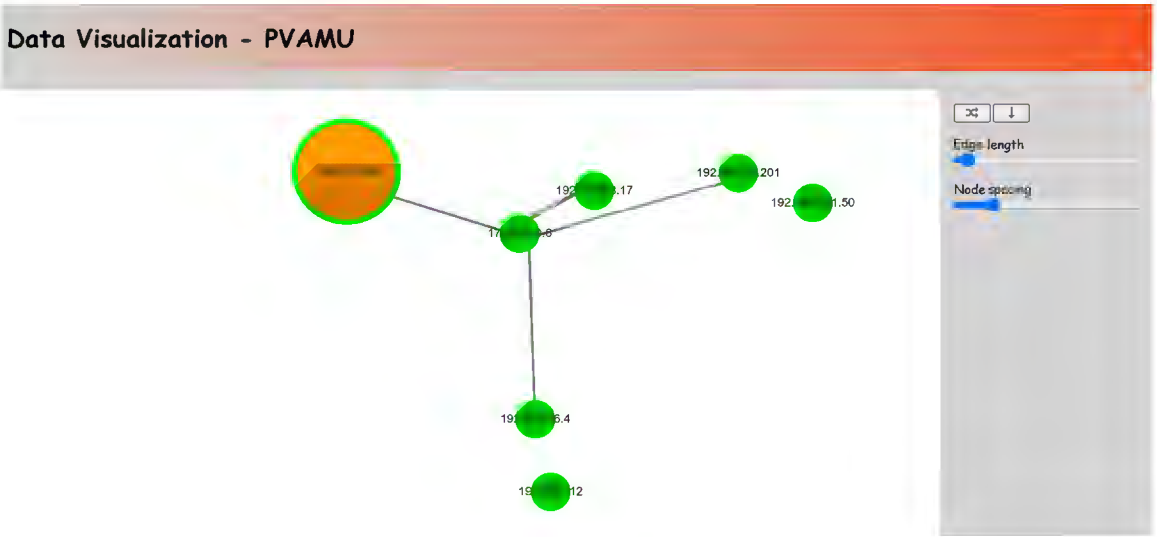
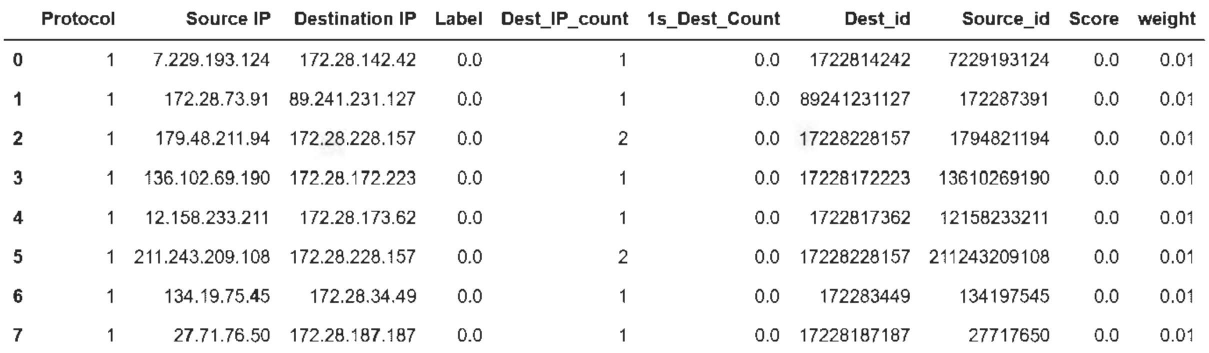


Fig. 3.6 Visualization representation Nodes and Edges between the nodes.



	Protocol	Source IP	Destination IP	Label	Dest_IP_count	1s_Dest_Count	Dest_id	Source_id	Score	weight
0	1	7.229.193.124	172.28.142.42	0.0	1	0.0	1722814242	7229193124	0.0	0.01
1	1	172.28.73.91	89.241.231.127	0.0	1	0.0	89241231127	172287391	0.0	0.01
2	1	179.48.211.94	172.28.228.157	0.0	2	0.0	17228228157	1794821194	0.0	0.01
3	1	136.102.69.190	172.28.172.223	0.0	1	0.0	17228172223	13610269190	0.0	0.01
4	1	12.158.233.211	172.28.173.62	0.0	1	0.0	1722817362	12158233211	0.0	0.01
5	1	211.243.209.108	172.28.228.157	0.0	2	0.0	17228228157	211243209108	0.0	0.01
6	1	134.19.75.45	172.28.34.49	0.0	1	0.0	172283449	134197545	0.0	0.01
7	1	27.71.76.50	172.28.187.187	0.0	1	0.0	17228187187	27717650	0.0	0.01

Fig. 3.7 Pre-processed final dataset overview.

Methodology and Experimental setup (Back-end) for ML and DL Models

DL and ML algorithms such as SVM, Decision Tree, and Deep Neural networks are used as baseline algorithms for predictive analysis. In addition to the conventional methods of prediction, analysis, and comparative assessment of these three models using an evaluation matrix, a novel and innovative approach introduces a naive visualization tool. This kind of approach has not been explored in depth before. In this case study, more focus was projected on the analysis by visualization tool rather than the evaluation matrixed-based method. The necessary background of these three algorithms is given in Section 1.2 in detail. The following section focuses on implementation details and the flow of the model.

Decision Tree

In this venture, the basic classification model of the Decision Tree is called from the ski-learn library. The coding part has been done using Python programming. Section 1.2.3 described the model background and details in-depth [40,41,42,44]. A simple baseline model was used here with no pruning methods. A flow diagram is shown in Fig. 3.8. Future research works could consider improving the model's accuracy using different techniques. In normal practice, a single dataset is taken and split with 70% of test data assigned for training purposes and 30% for testing. This venture used two separate datasets for training and testing. Firstly, training data was stored in the cache, and the model was trained with the training dataset in the initialization part. Then, test data was applied to the pre-trained Decision Tree model to predict attacked nodes. More fields were added in the feature engineering process and described in Section 3.3.2. The model's output was in python format and needed to be converted to JSON format so that the visualization tool could be used as input data. The flask API handled this work. Further detailed information is given in Section 3.6.

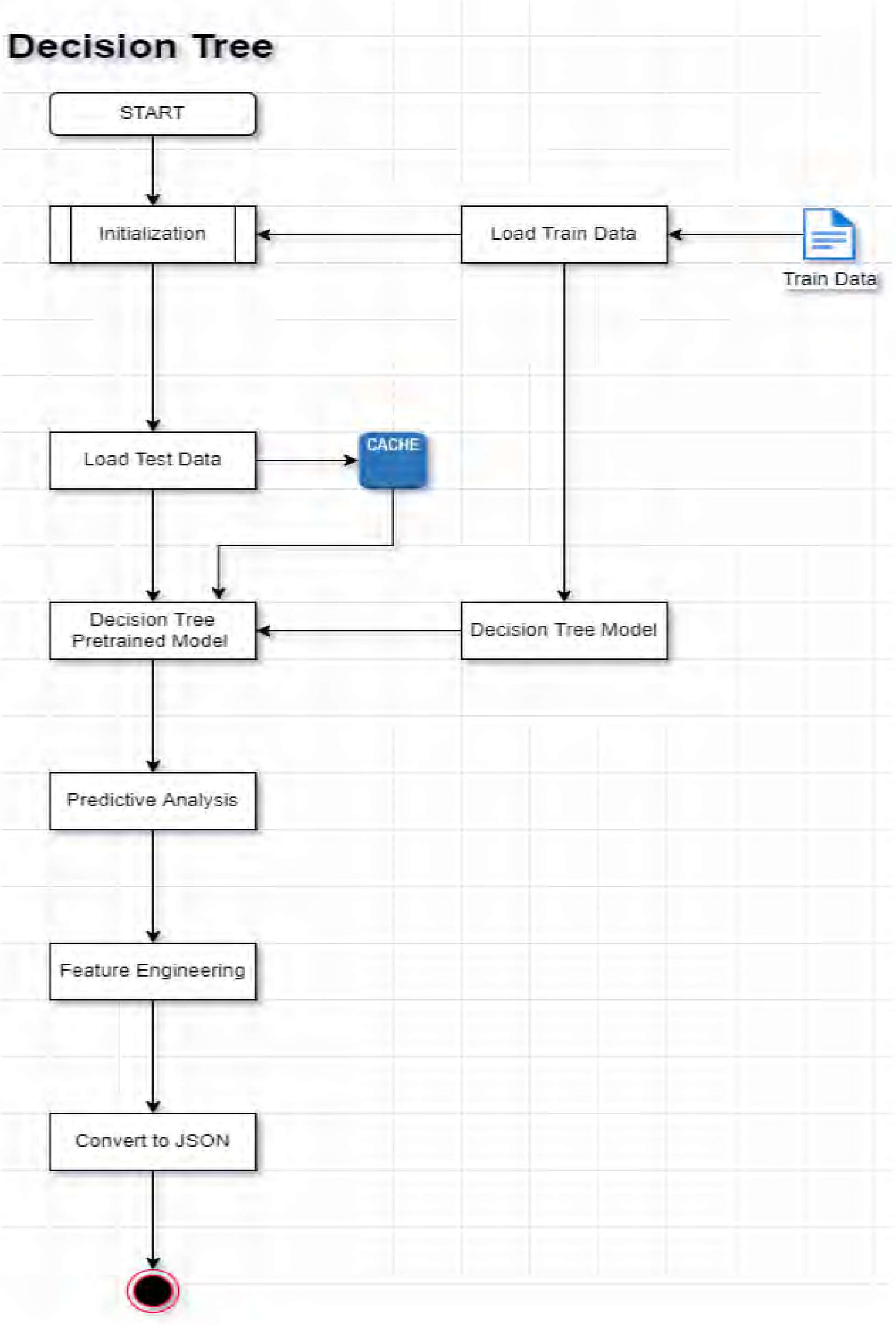


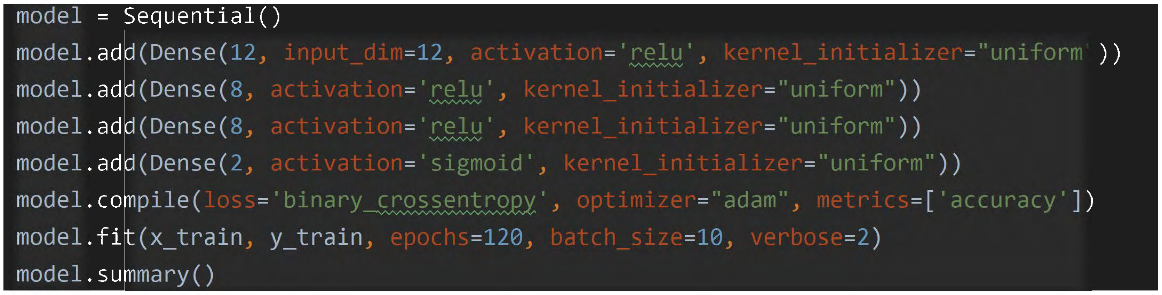
Fig. 3.8 Flow Diagram of Decision Tree Model.

SVM

Support Vector Machine is very popular in classification problems. More information is provided in Section 1.2.2. The model has been called from sklearn.svm library. In SVM, Kernel is the key performer for the selection of hyperplane, and the details are provided in [34,35,36] Section 1.2.2. A flow diagram is shown in Fig. 3.9. The total flow is like the Decision Tree and mentioned in Section 3.4.1

DNN

In the DNN model, four layers were used, including an input layer, two hidden layers, and an output layer. The loss function was binary_crossentropy, and the optimizer used was Adam. Two activation functions were used. The first was ReLu, and for the output layer, it was Sigmoid. The Sigmoid activation function was used because the desired output was in binary form output of either 1 or 0. An in-depth explanation is given in Section 1.2.4. The model used here is sequential, and the design of layers is shown in Fig. 3.10, and the flow diagram is shown in Fig. 3.11



```
model = Sequential()
model.add(Dense(12, input_dim=12, activation='relu', kernel_initializer="uniform" ))
model.add(Dense(8, activation='relu', kernel_initializer="uniform"))
model.add(Dense(8, activation='relu', kernel_initializer="uniform"))
model.add(Dense(2, activation='sigmoid', kernel_initializer="uniform"))
model.compile(loss='binary_crossentropy', optimizer="adam", metrics=['accuracy'])
model.fit(x_train, y_train, epochs=120, batch_size=10, verbose=2)
model.summary()
```

Fig. 3.9 DNN Model Design.

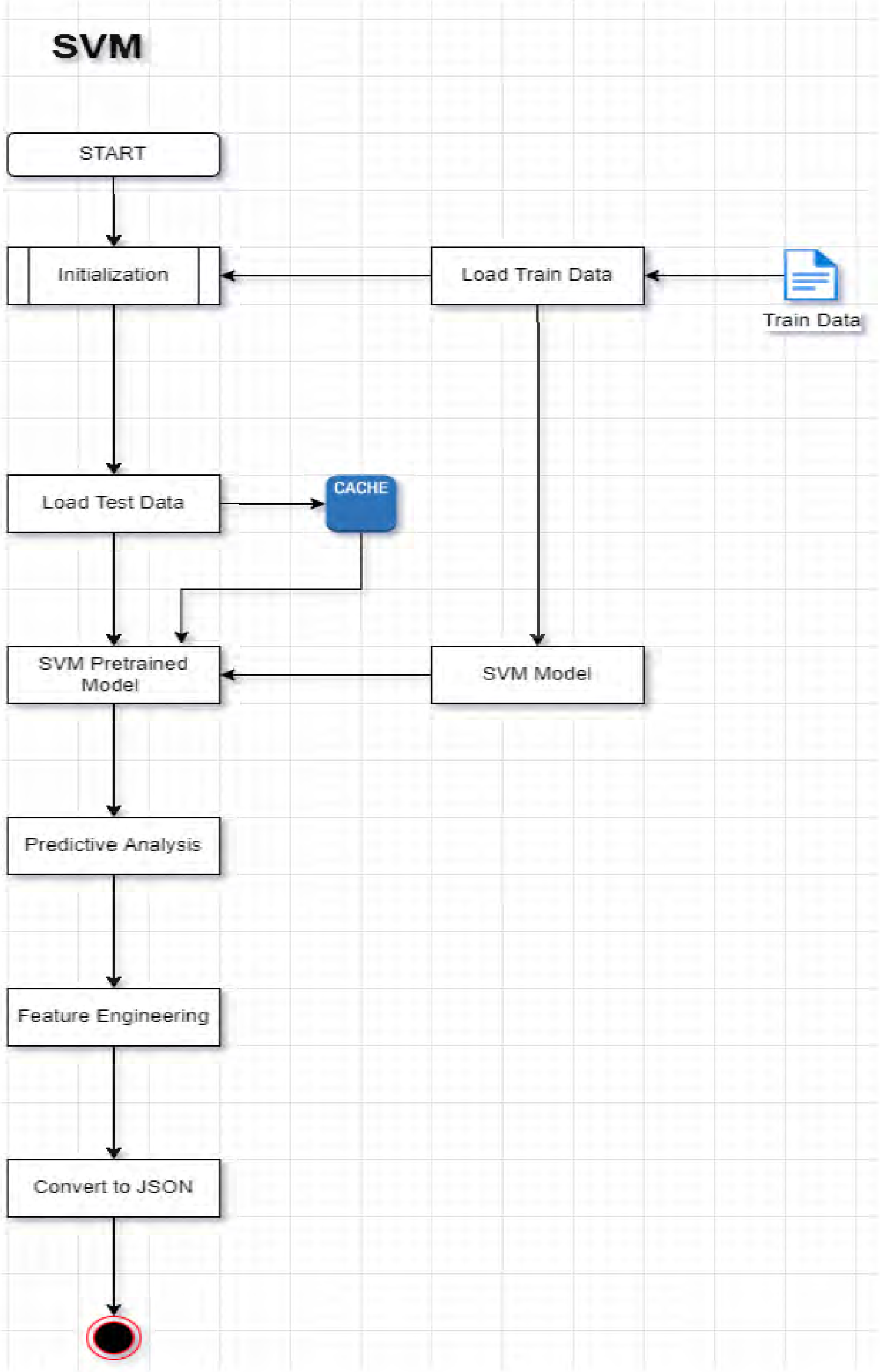


Fig. 3.10 Flow Diagram of SVM Model.

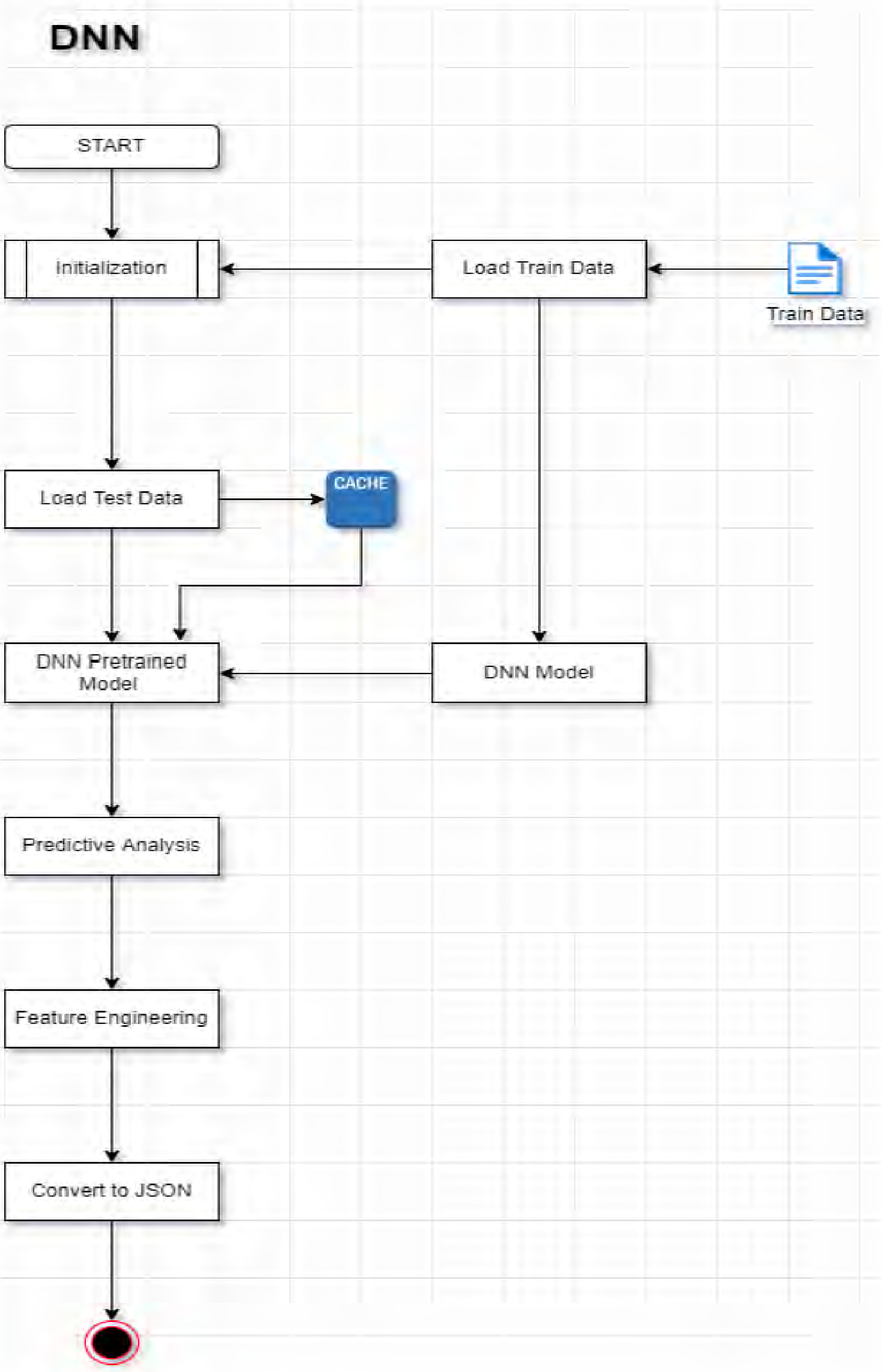


Fig. 3.11 DNN Model Flow Diagram.

Methodologies used for Visualization (Front-end User Interface) for Network Monitoring

Along with the predictive analysis for the detection of cyber-attack, a graphical user interface (GUI) helps visualize the attack in a timely manner rather than going back and forth in reports to find the attack node. In this research, a novel approach was opted for finding attacked nodes through a frontend tool. Cytoscape visualization tool libraries based on JavaScript were carefully chosen to project the GUI for the tool. Core app plugins are modified according to the requirement and implemented on a web-based portal, meaning they were hosted as a website for accessing visualization tool, as shown in Fig. 3.12.

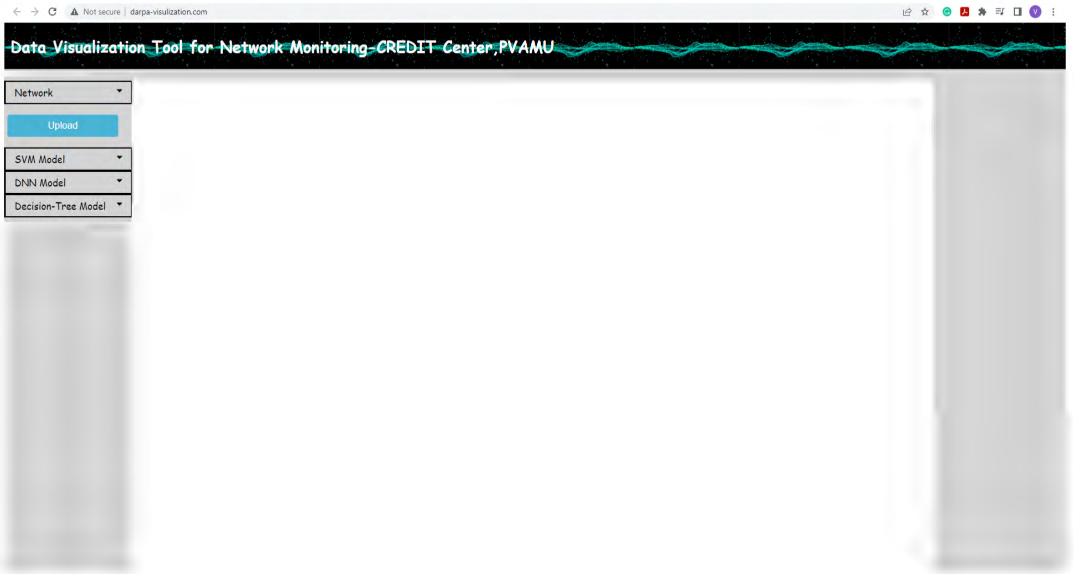
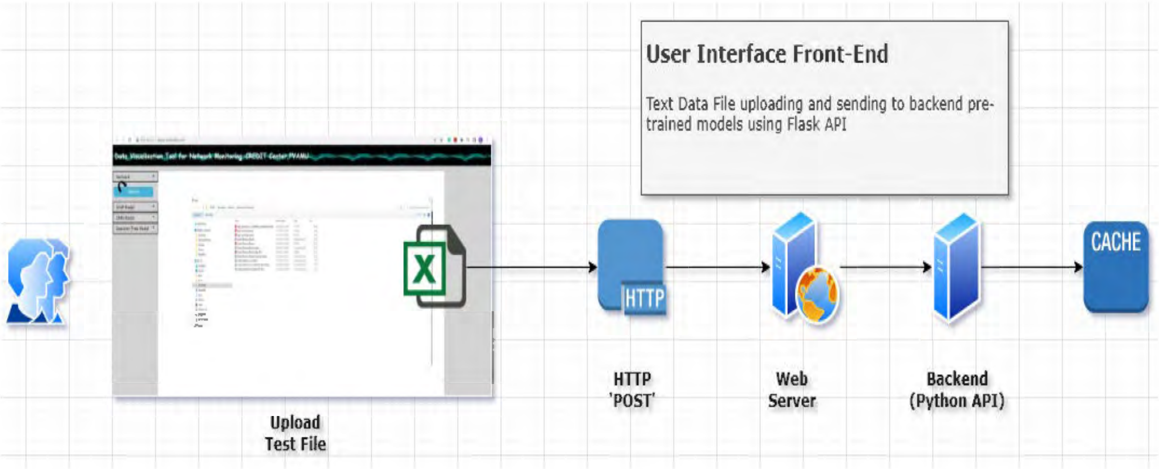

Not secure | darpa-visulization.com
Data Visualization Tool for Network Monitoring-CREDIT Center,PVAMU
Network
Upload
SVM Model
DNN Model
Decision-Tree Model

Fig. 3.12 Visualization Tool website.

The visualization tool operates within a Client-Server Architecture, where the frontend (client) employs an HTML-based User Interface (UI), and the backend (server) is powered by a Python-based API housing the essential domain logic.

Frontend (UI): Frontend comprised HTML, CSS, JavaScript, IIS, and Cytoscape library to visualize produced data by core domain which is explained in detail in the following sub-sections. Usually, HTML, CSS, and JavaScript are bundled together to create a webpage [84]. A web server in a big company scenario is a separate server appointed for only web-related databases and queries. Here, a computer has been hosted as a web server using IIS. When the user first lands on UI, they are asked to upload a Test Data file. With an HTTP "Post" request, they are sent the contents of that file to the backend (server), where it gets stored in cache memory. Further, has been used for testing with pre-trained ML and DL models. As shown below in Fig. 3.13, the sample file was used as input test data and can be used for all three models.



Text Data File uploading and sending to backend pre-trained models using Flask API

Fig 3.13 Test File uploading from Frontend.

Backend: When the user wishes to process the provided Test file by a particular model like Decision Tree, SVM, and DNN, the client's request gets accepted and carried out the following steps as per the selected model. For detailed descriptions of the process or flow of respective models, see Sections 3.4.1, 3.4.2, and 3.4.3 flow diagrams.

IIS Webserver

IIS stands for Internet Information Services. It could turn a computer into a web server. This web server can be used to provide world wide web services along with other additional services like FTP, SNMP, and NTP. IIS can host and manage websites. Once After acquiring the initial IP address, the next step involves registering the domain on a DNS server and ensuring proper network configuration [80]. IIS works on client and server architecture where any web browser like internet explorer works as a client and requests files from the IIS Web server (refer to Fig. 3.14). The IIS web server transfers the requested file and headers containing connection information. HTML tags in the header of the file help to format and display correct data in the file. The website URL is defined as "www.darpa-visulization.com, "and configuration details are provided in Fig. 3.15. Microsoft IIS configuration document [81] was used as a reference for the configuration.

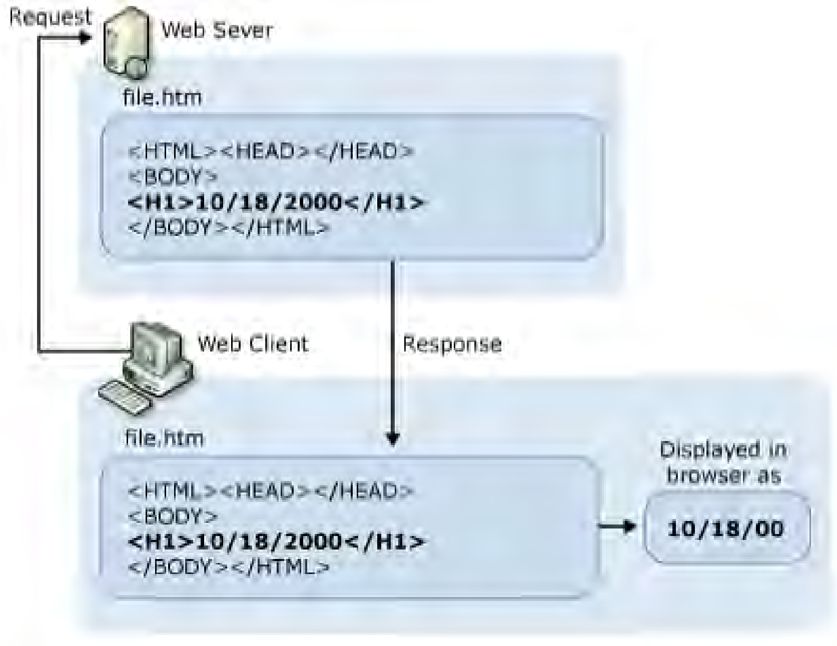
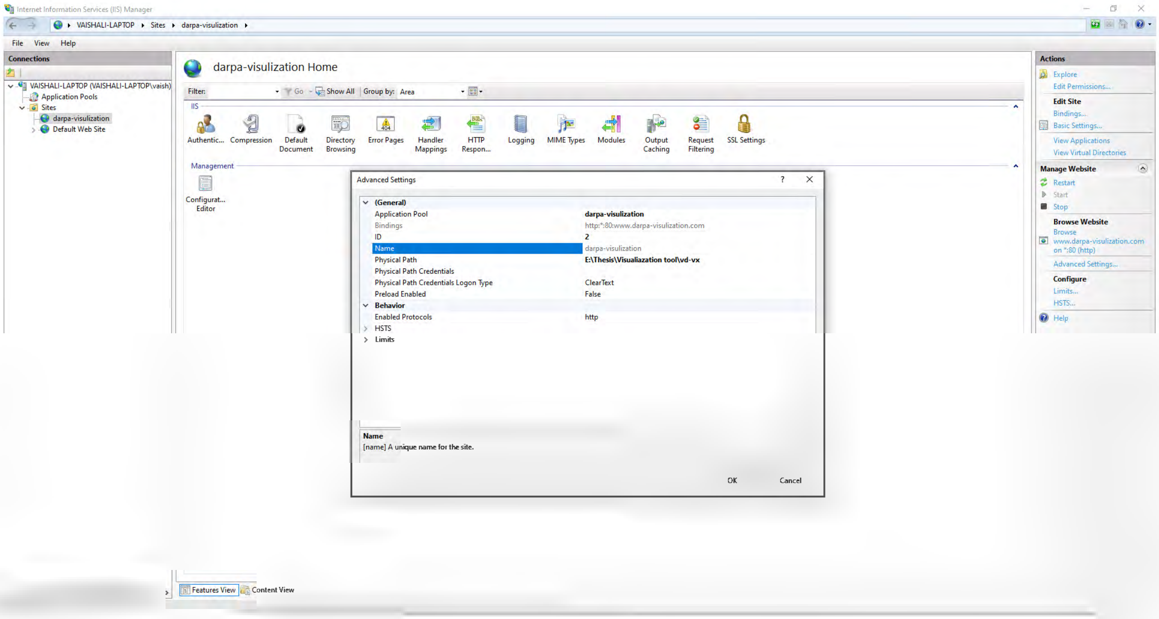


Fig.3.14 IIS Client and Web server architecture [80].



Advanced Settings

(General)

Application Pool	darpa-visulization
Bindings	http:*:80:www.darpa-visulization.com
ID	2
Name	darpa-visulization
Physical Path	E:\Thesis\Visualiazation tool\vd-vx
Physical Path Credentials	
Physical Path Credentials Logon Type	ClearText
Preload Enabled	False

Behavior

Enabled Protocols	http
HSTS	
Limits	

Name
[name] A unique name for the site.

OK Cancel

Fig. 3.15 Internet Information Service Manager Configuration.

Within the IIS environment, the DARPA visualization site was allocated the HTTP port number 80. An IP address could be manually assigned or left to default configuration.

HTML

HTML stands for "Hypertext Markup Language." It specifies web page structure, or in simple language, it is called building blocks of the web. HTML code was used in this project to structure a web page and its contents. HTML code consists of a series of elements that make the page a certain way it should appear, and the browser understands those markups and display accordingly. Hypers links are also an essential aspect of HTML programming, making the web an actual web. All this basic information about HTML is mentioned in the document [82]. The setup of HTML element tags, structure, and formatting for "www.darpa-visualization.com/" adheres entirely to the configuration detailed in the document [83] (see Fig. 3.16).

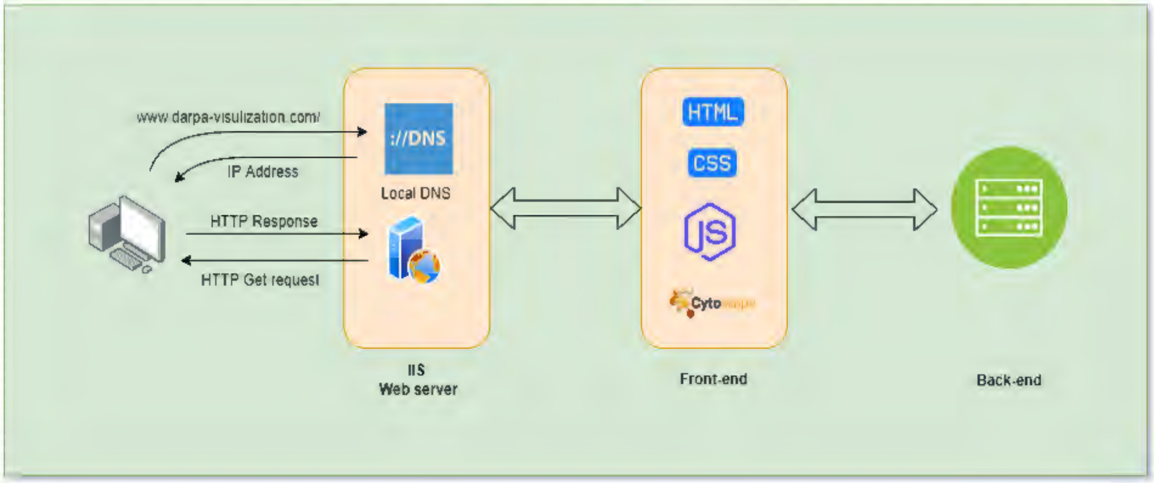


Fig. 3.16 “www.darpa-visulization.com/” URL flow.

The flow after URL button is pressed-briefly described.

1. Enter a URL into a web browser
2. The browser looks up the IP address for the domain name via Local DNS. In this case, it is the IIS Web server.
3. Gets response with IP address
4. The browser sends HTTP requests to the web server.
5. The server sends back HTTP response
6. The browser begins rendering the HTML
7. The browser sends requests for additional objects embedded in HTML (images, CSS style files, JavaScript, Cytoscape libraries) and repeats steps 3-5
8. Once the page is loaded, the browser sends further sync requests as needed.

After getting a response from the server, the web page displays Below, it explains the configuration and designed part using HTML along with CSS styling (refer to Fig. 3.17).

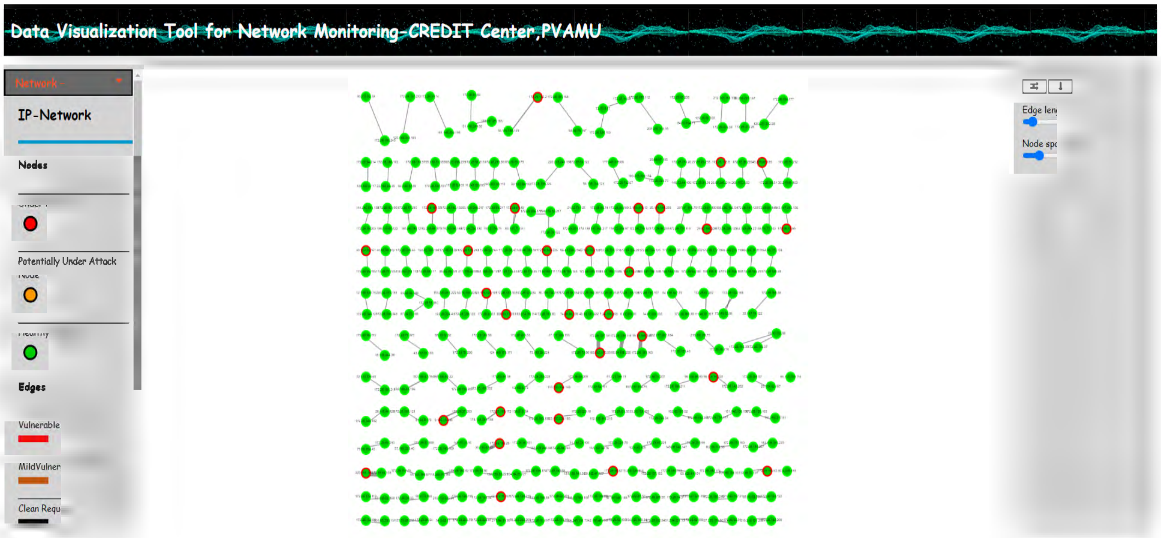


Fig. 3.17 Visualization Tool with HTML and CSS coding.

In Fig. 3.17, at the top bar, it is written "Data Visualization Tool for network monitoring for CREDIT Center, PVAMU." On the left-hand sidebar where the network tab is seen, the representation is done using HTML and CSS code. For all color coding and meaning of Nodes and Edges explained in Section 3.2.2, refer to Figs. 3.4 and 3.5. In Fig. 3.18. on the lefthand sidebar, the Decision Tree Model, SVM Model, DNN Model and Upload, and Analyze tabs are all coded or built with the help of CSS and HTML designing [82,83]. Upload and analyze are the processes connected to the backends and the click's frontend sends the requests to the backend endpoints. This process is explained in the proceeding sections. In the exact figure, the classification description includes actual and predicated information, representing the naïve method of predication analysis of modeling results, and it is explained in detail in Chapter 4, performance analysis.

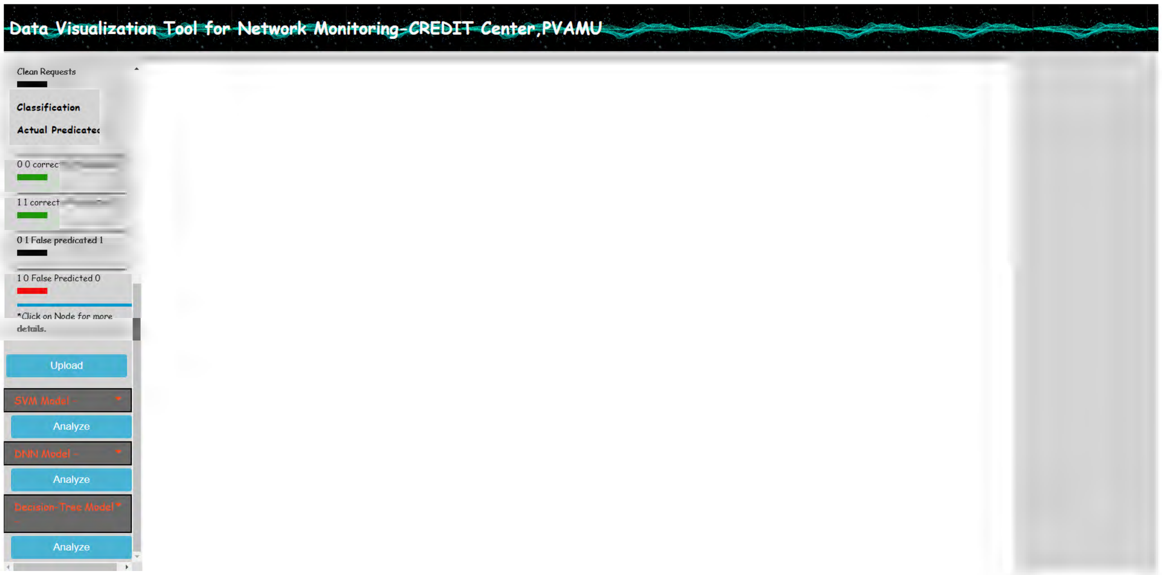


Fig. 3.18 Visualization Tool Left panel with HTML and CSS coding.

CSS

Cascading Style Sheets (CSS) is a stylesheet language in web development. CSS is used in documents written in HTML or XML. It designates the element's presentation on screen or paper or in speech. This venture has used CSS and the HTML language to develop the web page by following the building block of CSS documentation [85,86]. Web development work is collectively done using HTML and explained in section 3.5.2.

JavaScript

JavaScript is a scripting programming language. It adds interactivity to the website. CSS, HTML, and JavaScript go stack over stack to make the website or web development dynamic. With the help of APIs (Application Programming Interface), JavaScript dynamically modifies the CSS and HTML code accordingly. For more details, refer to the [87] website for the programming steps for JavaScript.

Cytoscape

Cytoscape is an open-platform visualization library based on Java and JavaScript programming languages. It is nothing but a graph theory library to visualize relational data such as, biological data or social networks' graphical representation. Cytoscape library was designed to make it as easy as possible for programmers to use graph theory in their projects, whether for server-side analysis in a Node.js app or for a rich user interface [88]. Here, the dynamic functionality of this library was used by modifying the code pursuant to the requirement to display the IP network nodes and their connectivity with each other. Lines were used to display the IP traffic between the nodes. Cola.js plugin graph representation has been used among different interfaces provided by the tool providers. These libraries are mainly used to represent the biomolecular interactions in many projects. However, a naïve approach was chosen for this research work and used to represent the IP network. Code and more graphical plugins can be found in [88].

Application Programming Interface (API) Methodologies

The below flow diagram depicts the end-to-end process and architecture of the loosely coupled Visualization tool in Fig. 3.19. Here, the backend the mythologies used are Python programming language and Flask API.

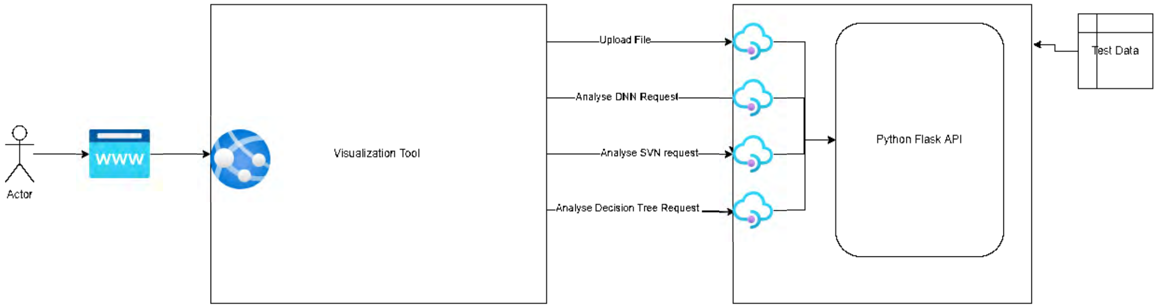


Fig .3.19 Architecture diagram depicts the end- to-end process.

Python

Python programming language is prevalent among Data Scientists. It has the advantages of simple statistical data cleaning and pre-processing functions and inbuilt libraries like Pandas, Numpy, sea-born, Scikit- learn, and Keras for a data science project. It makes the programming easy and less complicated to use. In this work, Python was used for programming all the backend ML modules DNN, Decision Tree and SVM. All the syntax and information for Python are referred from [89]. Python provides one of the best APIs like Flask – API or Django-API for connecting the backend with web applications. These API functions with CSS, HTML, and JavaScript provide dynamic functionalities to web applications. Information about Flask API is provided next.

Flask API

Application Programming Interface (API) is an interface used to communicate between two applications, the internet, and server. For example, the application connects to the internet and sends data to the server. The server receives the data, interprets it, performs the necessary action, and then sends it back to the application securely. In this venture, the REST API was used to connect to the web browser. Web API uses HTTP request methods and defines the structure of response messages in web communication.

These response messages are preferred in XML or JSON formats because they are easy for applications to manipulate.

For developing web applications, the choice for API is mainly between the two frameworks, Flask API, and Django API. Every API has pros and cons. Before selecting the API framework, all the pros, cons, and requirements were taken into consideration [90]. REST API has a comparatively simple framework and is more flexible and compatible with python applications than the Django API. It can be changed pursuant to needs. On the other hand, Django is more rigid, fits under one umbrella, and cannot be changed. Mainly, Django has been used in complex projects. Flask is a simple, decoupled, and fastest API for small web applications, used in Facebook/Twitter bots and a few more social networking sites. The only disadvantage Flask API has is that it has a singular source, meaning it handles each request one after the other. Implementation of flask API is provided in [91].

Final Experimental Setup

After looking into all the front-end and back-end methodologies, the final setup and process diagram are shown in Fig. 3.20. Every aspect of the diagram is explained separately in Sections1 to 3.6.

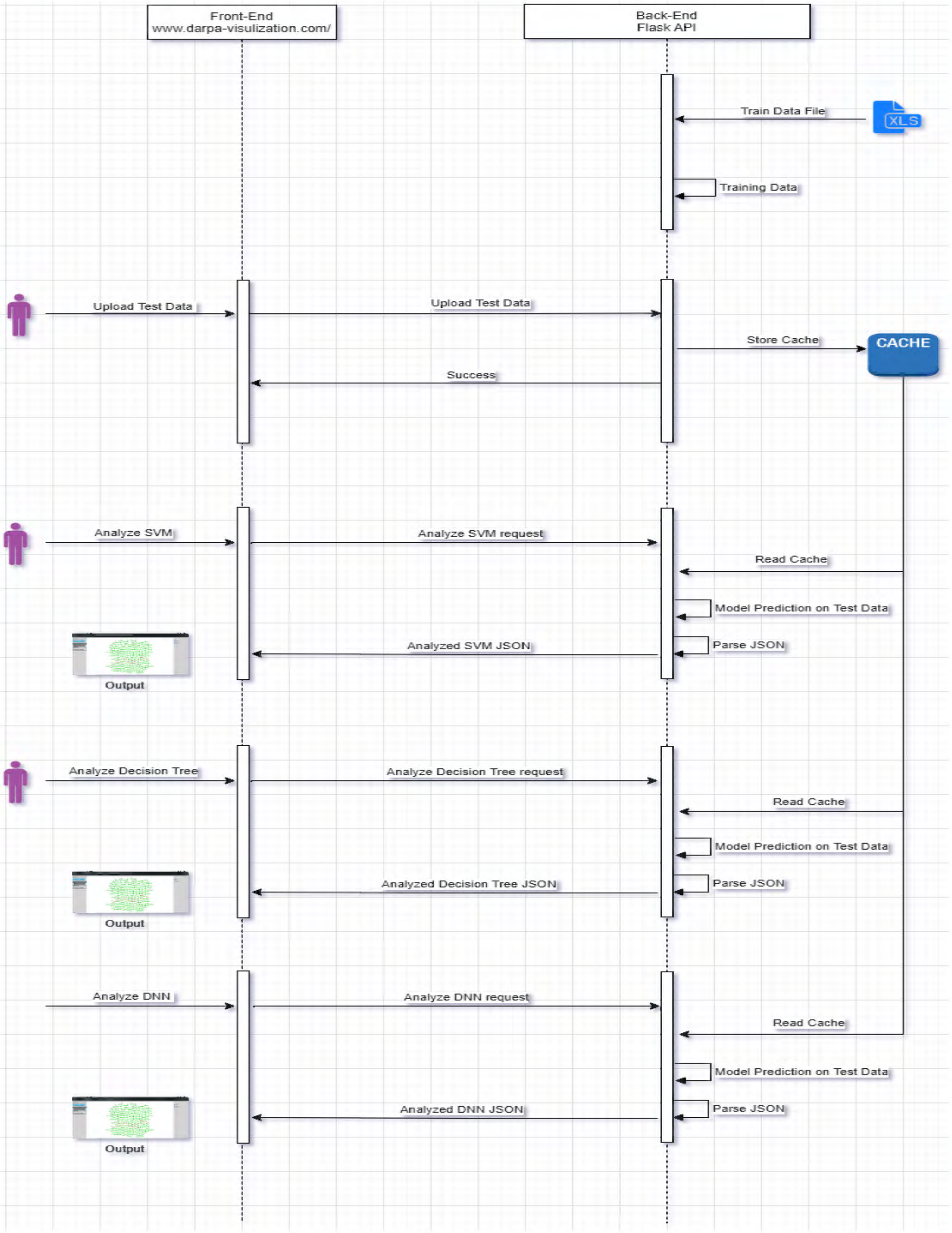


Fig. 3.20 Final Experimental setup and process diagram.

IV. PERFORMANCE ANALYSIS

The proposed models were evaluated on the "DARPA_2009_DDoS_attack-20091105" dataset provided by the LANDER project and described as an SYN flood DDoS attack metadata. This dataset was part of the 2009 DARPA Scalable Network Monitoring (SNM) Program Traffic. Dataset is labeled dataset and comes under Supervised learning. The dataset details were described in Section 3.2, and all preprocessing and feature engineering work were explained in Sections 3.2 and 3.3.

The dataset file contains six minutes of IP network traffic between nodes. Few IP nodes are attacked with sync flood attack messages generated by illegitimate users intended to harm the network. All background about DDoS attacks and cyber-attacks was provided in Sections 1.1.1 and 1.1.2. The advantage of this tool was that the dataset was replaceable with some other type of network attack dataset with minimal changes at the backend, preprocessing, and feature engineering process. The dataset from DARPA had 1 M records of TCP-IP messages between the nodes.

Three baseline models were used to perform predictive analysis on node attacks viz Decision Tree, SVM, and DNN for all experimental setup and methodologies on models. Refer to Section 3.4.1, Section 3.4.2, and Section 3.4.3, respectively. 1M dataset file named "DARPA_Fullset" was used to train the models, and then 630 records "DARPA_sample_1" was used for testing the model performances. The small size of is test file involved consideration of two factors:

i. There is no doubt about the Cytoscape visualization tool scalability to represent the number of nodes. However, this project was performed on Intel(R) Core (TM) i7-8565U CPU @ 1.80GHz 1.99 GHz with RAM 16.0 GB (15.7 GB usable), with primary graphic card: Intel(R) UHD Graphics 620 which is not sufficient to represent to display the enormous data graphically.

ii. Using GPU or Linux machines was not an option because Cytoscape is incompatible with the Linux systems and needed any GUI.

Hence, the test dataset needed to be small to represent predicted attack data in the visualization tool GUI.

This study encompassed four distinct experiments. The initial experiment involved visually representing the dataset without engaging in model training or testing—a straightforward data presentation in a GUI. The second experiment entailed training a Decision Tree model using 1 million records and subsequently evaluating it with test records. The third experiment involved utilizing the SVM model to predict attacked nodes, employing the identical training and testing dataset. The fourth experiment incorporated the DNN model using the datasets. Finally, the performance evaluation of these models was undertaken using two distinct approaches, which are elaborated upon in the subsequent section.

Evaluation Methods

This research demonstrated two assessment methodologies: the initial method employs conventional evaluation metrics, while the second employs the Network Monitoring Tool. The subsequent section provides a concise overview of both techniques, with a particular emphasis on the Visualization tool approach.

Traditional Evaluation Metrics

Many researchers have taken advantage of evaluation metrics in the past years successfully. Likewise, [92] authors X. Dong, U. Victor, and L. Qian employed accuracy, precision, Recall, and F-score evaluation techniques to check the performance of models for fake news. Likewise, within this undertaking, evaluation metrics were employed to assess the performance of the baseline model. Before delving into the outcomes of the model, a description of the evaluation metrics is provided [92]. Accuracy, denoted as the ratio of correctly predicted records to the total number of records, is elaborated Refer to Equation 4.1.

$$Accuracy = \frac{N_{correct}}{N_{total}} \qquad (4.1)$$

Here, accuracy is calculated by dividing the number of correctly detected records by the total number of IP records between the nodes. The task is a binary classification task, so F-score values employ the performance for each class. F1 score is the harmonic mean of precision and Recall. It offers a deeper understanding of the dataset's dynamics, particularly when there is an imbalance, and sheds light on the model's ability to make precise predictions without favoring the majority class of records. Refer to Equation 4.2.

$$Fscore = \frac{2*Precison*Recall}{Precision+Recall} \qquad (4.2)$$

Model Accuracy is inclined toward the prediction of the majority class, that time F1 score gives a good prediction. F1 ranges from 0 to 1, considering 1 is a perfect score, and the model predicts each observation correctly; 0 is the worst score. Prior to delving into Precision and Recall, shed light on fundamental concepts pertinent to this research, which encompass TP, TN, FP, and FN. This dataset has two classes "attacked nodes – 1" and "Not-attacked nodes – 0."

TP – Number of IP nodes are attacked and predicated as attacked by model.

TN - Number of IP nodes not attacked and predicated as not attacked by model.

FP - Number of IP nodes not attacked but predicated as attacked by model.

FN - Number of IP nodes attacked but predicated as not attacked by model.

Therefore, precision pertains to the total number of accurate predictions made for attacked nodes. Refer to Equation 4.3.

$$Precision = \frac{TP}{TP+FP} \qquad (4.3)$$

In Recall, the actual truth is a baseline. In this case, the total number of attacked nodes from the dataset. Hence, Recall reflects the accuracy of our correct predictions for attacked nodes out of the total true instances. Refer to Equation 4.4.

$$Recall = \frac{TP}{TP+FN} \qquad (4.4)$$

Accuracy is the standard evaluation metric across many projects, however, here, this research considers F1-Score as an evaluation for the performance of models as the dataset is not balanced.

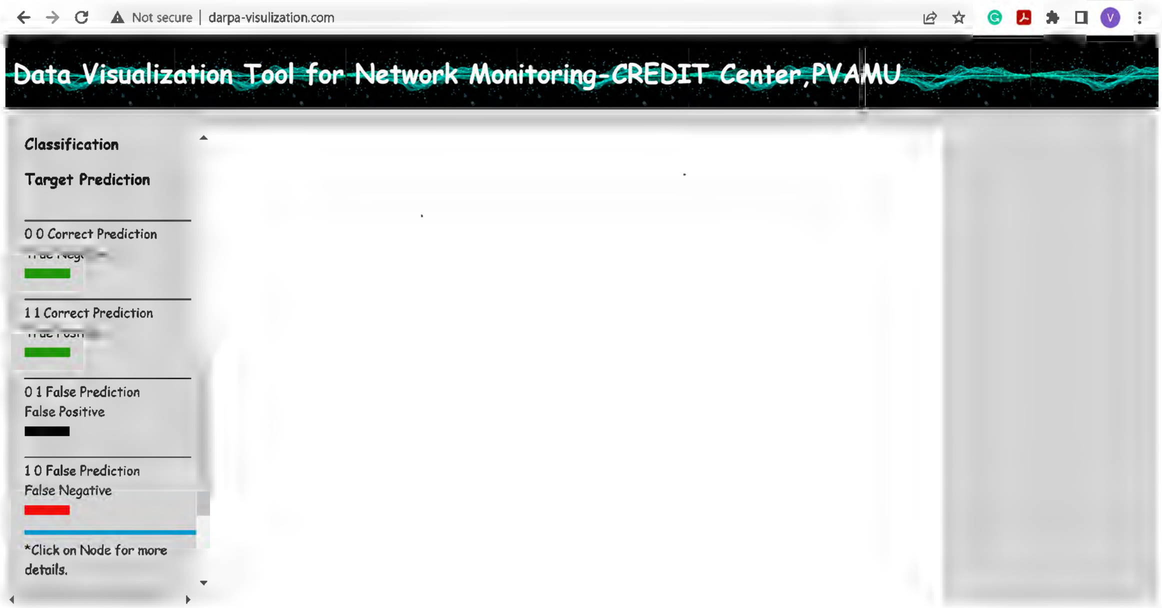
Evaluation with Network Monitoring Visualization Tool

This study explored the additional and naïve evaluation method for analyzing the prediction results of models through a visualization tool. A network monitoring tool is mainly developed to monitor the cyber-attacks on the network so the user of the tool can easily see the attacked nodes and take the appropriate action to avoid damage to the network. The traditional approach using evaluation matrices provides the accuracy and F1-score of the model. Using this information, one can improve the model's predictions and make modifications so models can predict the attacks better. Nonetheless, this traditional analysis approach is not helpful in conditions where the user can take necessary actions according to the predictions on time. One fundamental but essential question arises, whether the attack predictions displayed on the IP nodes in the form of color nodes (explained in section 3.2.2) on the network GUI were true compared to the ground truth. How can the correctness of the evaluation conducted by the network monitoring tool be substantiated? This was achieved by incorporating functionality in both the backend and frontend code, comparing the ground truth (target column) with the predictions and displaying the outcomes in a color-coded format on the node. A detailed explanation is provided below and explained with the help of Table 1.

TABLE 1
GROUND TRUTH VS PREDICTION ANALYSIS TABLE

Ground Truth	Predictions	Analysis	GUI Representation of circle around the node
0	0	True Negative	🟢
1	1	True Positive	🟢
0	1	False Positive	⚫
1	0	False Negative	🔴

The comparison chart between ground truth and predicted values is displayed to show the results of predictions on the GUI screen above. It is represented in the form of the ring around the IP node irrespective of the node's color. Ring color represents whether it is TP, TN, FP, or FN. The information is also available on the screen for easiness for the user, as shown in Fig. 4.1



Not secure | darpa-visulization.com

Data Visualization Tool for Network Monitoring-CREDIT Center,PVAMU

Classification

Target Prediction

0 0 Correct Prediction
True Neg
🟩

1 1 Correct Prediction
True Posi
🟩

0 1 False Prediction
False Positive
⬛

1 0 False Prediction
False Negative
🟥

*Click on Node for more details.

Fig. 4.1 Ground Truth vs Predicted Analysis on the Visualization Tool Screen.

Possible outcomes of analyzed predictions are explained below with the diagrams.

(a) True Positive and True Negative node analysis with (Non-attacked) Normal nodes and Mild vulnerable Nodes: normal node is shown with the green color, whereas the green ring around the node signifies the prediction is true, meaning ground truth and prediction is same. The orange node indicates the node is mild vulnerable, but the prediction is correct compared to the ground truth. Refer to Fig. 4.2

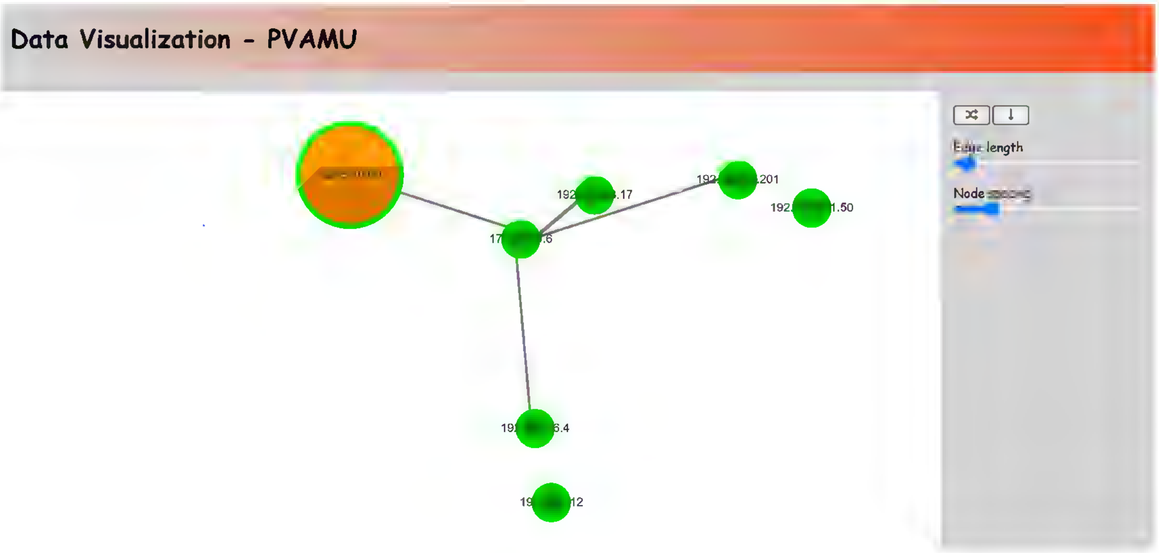


Fig. 4.2 True Positive and True Negative analysis with Normal and Mild vulnerable Nodes.

(b) True Positive and True Negative analysis with attacked Nodes: red nodes indicate they are under attack, and the green ring around it shows the prediction is correct compared with ground truth. Refer to Fig. 4.3

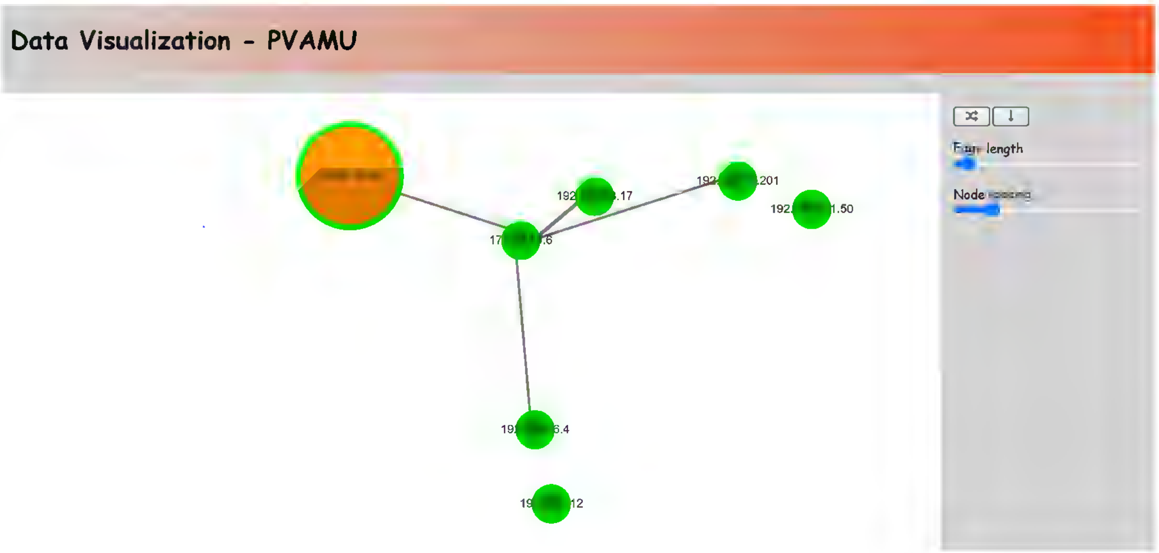


Fig. 4.3 True Positive and True Negative analysis with attacked Nodes.

(c) True Positive and False Positive analysis with attacked Node: Previously have seen the red node with a green ring, meaning the node is correctly predicted and under attack. Here, the black ring indicates the false positive, so the prediction is attacked, but the actual ground truth node is not under attack. Refer to Fig. 4.4

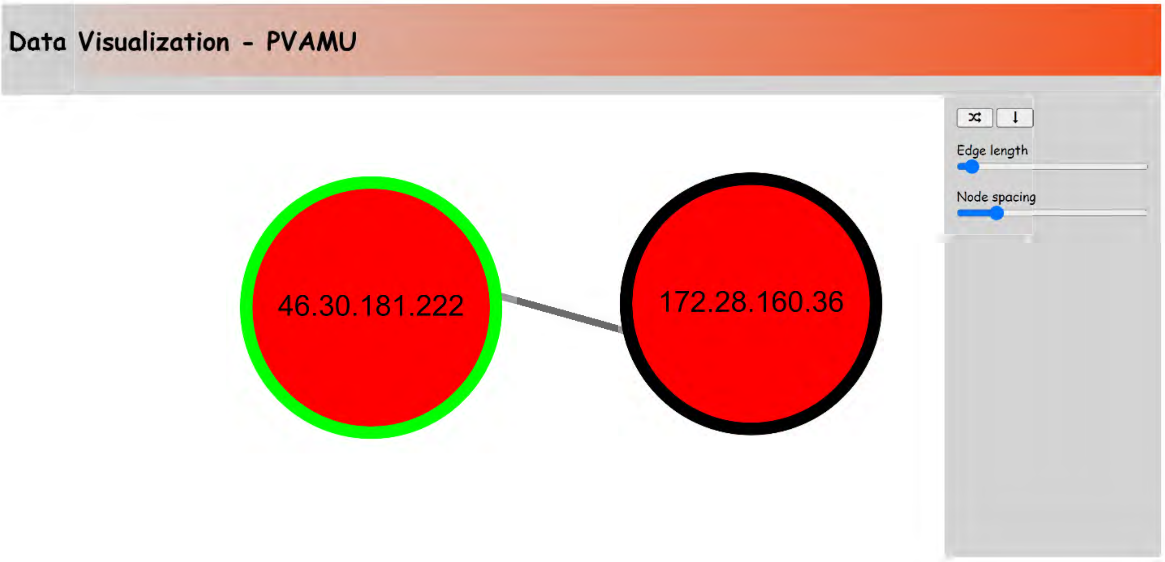


Fig. 4.4 True Positive and False Positive analysis with attacked Node.

(d) False Negative analysis with Normal Node: false negative implies that ground truth is under attack (1), but the prediction is not under attack, which is wrong. So, the red ring around the node implies the false negative prediction. Refer to Fig. 4.5

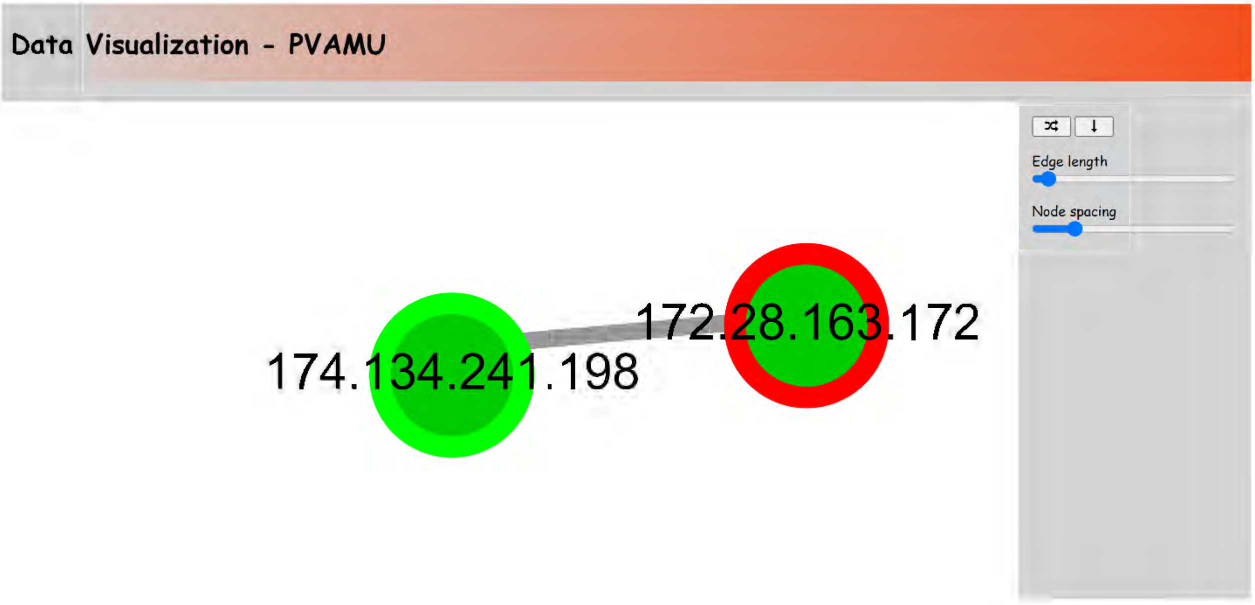


Fig. 4.5 False Negative analysis with Normal Node.

Section 4.2. elucidates the results obtained with both the evaluation methods one by one.

Result Analysis

Back-end

Table 2 shows the cyber-attack prediction evaluation results of the three models using the same labeled dataset for training and testing. SVM model shows promising results with 1M records of labeled data for predicting the attack. In contrast, the Decision Tree did not perform well in terms of accuracy compared to SVM and DNN with the same dataset size. Still, training time was much faster than in the SVM and DNN models, and it predicted the class 1 instances accurately compared to DNN and SVM. Accuracy vise DNN performed moderately better than the Decision Tree and SVM models but failed to predict the class 1 cases. From the experiment, it was observed that, with varying amounts of labeled dataset sizes, model performances also varied. When the labeled samples were increased, the DNN model performance also improved. With the results that Table 2. for the SVM model, it takes more time to train the model compared with the Decision Tree.

Despite good accuracy, normal SVM was not suitable for classifying a large dataset because SVM's training complexity is highly dependent on a large dataset. DNN model takes the highest training time compared with both models.

TABLE 2
DETAILED MODEL EVALUATION RESULTS GENERATED WITH LABELD DATA OVER DARPA DATASET

	Accuracy (%)	Classes	Precision (%)	Recall (%)	F-score (%)	Time
Decision Tree Model	0.80	0	0.89	0.88	0.89	2.56
		1	0.27	0.29	0.28	
SVM Model	0.86	0	1.00	1.00	0.93	116.88
		1	0.00	0.00	0.00	
DNN Model	0.87	0	0.87	1.00	0.93	1041.87
		1	0.00	0.00	0.00	

The primary aim of the research was to select three different models so they could accommodate the different sizes of datasets to get the best performances out of every model with respective data sizes, and it could be accredited as a robust and effective system. Still, there is a scope for improvement for class 1 (minority class) predictions caused due to the imbalanced dataset. Models' precision and accuracies were also high across the labeled data. After getting the predicted outputs from backend ML and DL models, the prediction results were pushed to the frontend via REST API to showcase predictions in the visualization tool.

Front-end

The Visualization Tool was built to show the backend modeling results of the predicted attack on GUI in an understandable format. To check the predictions of the test dataset on the frontend (visualization tool), the network engineer will first upload the test dataset file using the UPLOAD tab on the screen. The network topology of that dataset before going through any model prediction is shown in Fig. 4.6 None of the nodes within the tool exhibit colors or predictions on the display. This deliberate design facilitates the objective of comparing predictive analyses using ML and DL models, a goal that has been effectively realized

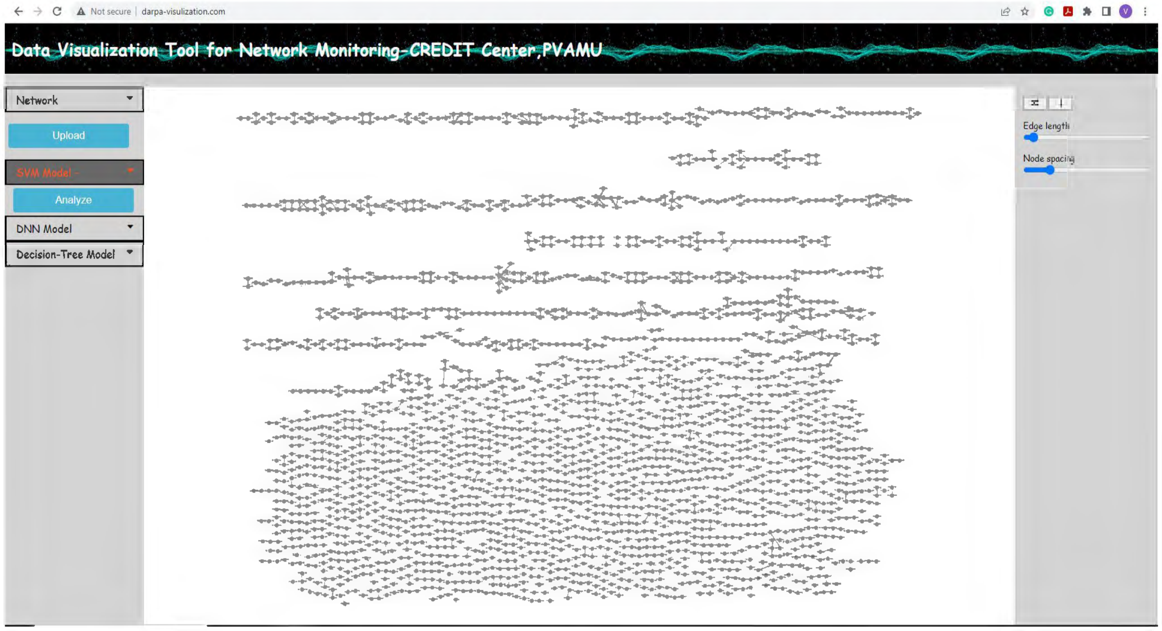

Not secure | darpa-visulization.com
Data Visualization Tool for Network Monitoring-CREDIT Center,PVAMU
Network
Upload
SVM Model
Analyze
DNN Model
Decision-Tree Model
Edge length
Node spacing

Fig. 4.6 Network topology of test dataset before prediction on visualization tool.

After that, the network engineer can choose any model to predict an attack by selecting the Analysis tab on the screen. Fig. 4.7 shows the attack prediction done by the Decision Tree model on network nodes and displayed on the visualization tool, after getting the results pushed from the backend pretrained Decision Tree model. The result evaluation of prediction on the test dataset was explained in detail in Section 4.1.2.

Here, it is evident how the backends precise outcomes were mirrored within the visualization tool, providing network engineers with convenient insights for prompt decision-making. Similarly, Figures 4.8 and 4.9 depict the results for the SVM model and

DNN.

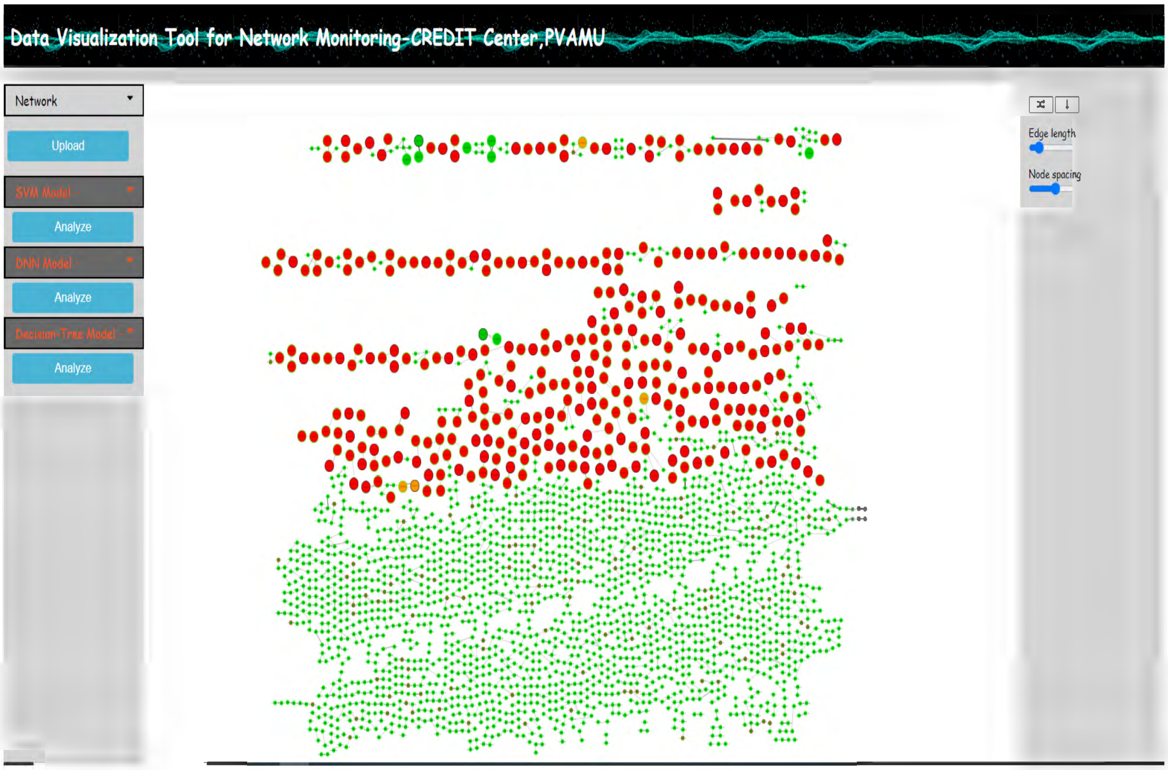


Fig. 4.7 Decision Tree model prediction on visualization Tool.

After comparison of results in Table 4.2 with the prediction output of the Decision Tree on the visualization tool in Fig. 4.7, we can conclude that the Decision Tree did a great job in the prediction of both the classes for 0 and 1, even though it predicted many records as false positive. Still, it did a decent job in predicting the minority class cases which were under attack and showed as false negative on the screen. All three models predicted the attacks and displayed them on the visualization screen. Refer Figs. 4.8 and 4.9.

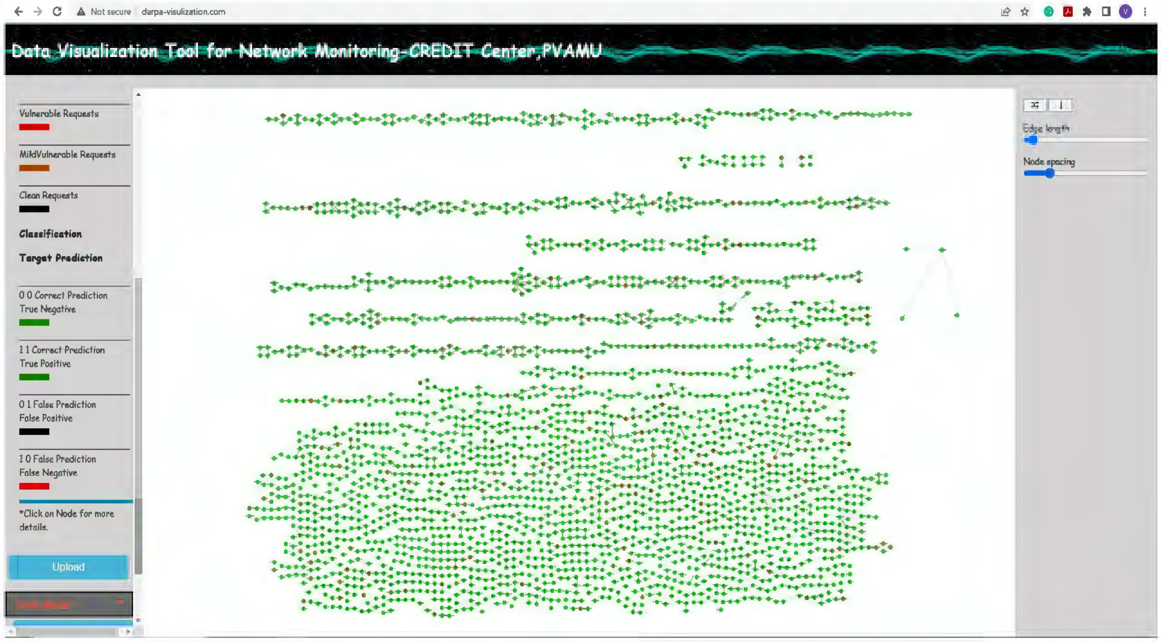


Fig. 4.8 SVM model prediction on visualization Tool.

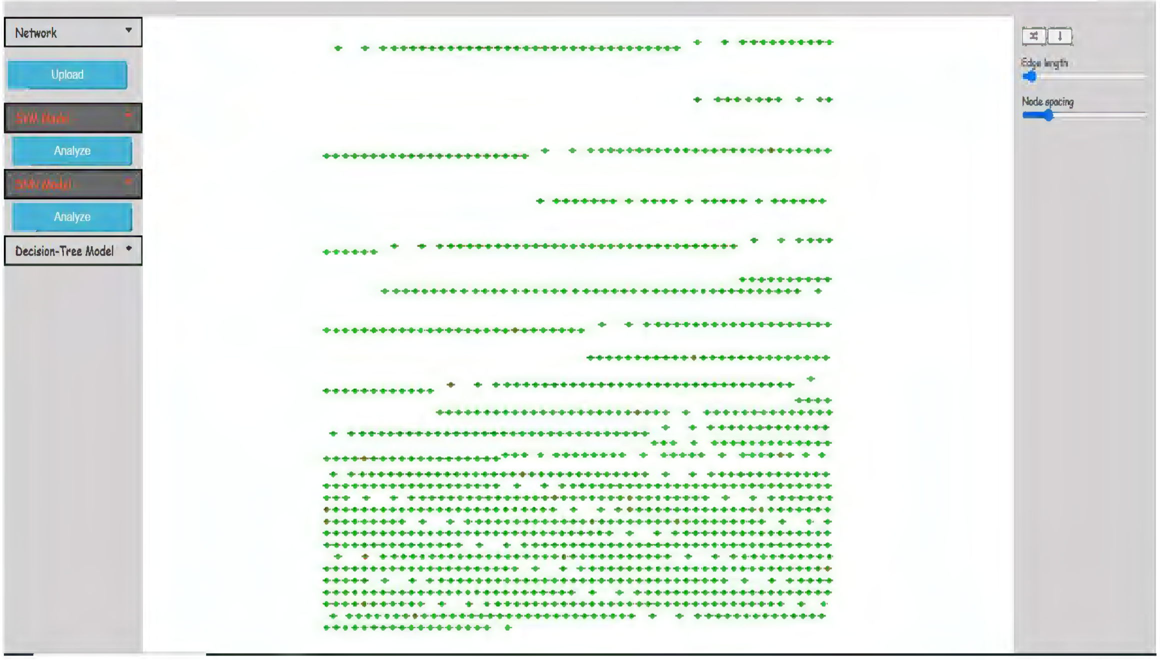


Fig. 4.9 DNN model prediction on visualization Tool.

Layered Network Topology

Layered network topology has been achieved, as shown in Fig. 4.10. The visualization tool's webpage was designed in such a manner when it double-click on any node on the GUI, it will display that node with all connected nodes automatically on the different webpage. Refer to Fig, 4.10, which is an output of the Decision Tree model’s prediction on a testing dataset. One node from the black circle node was selected and displayed all the connected nodes generated on a new webpage as shown in Fig. 4.11

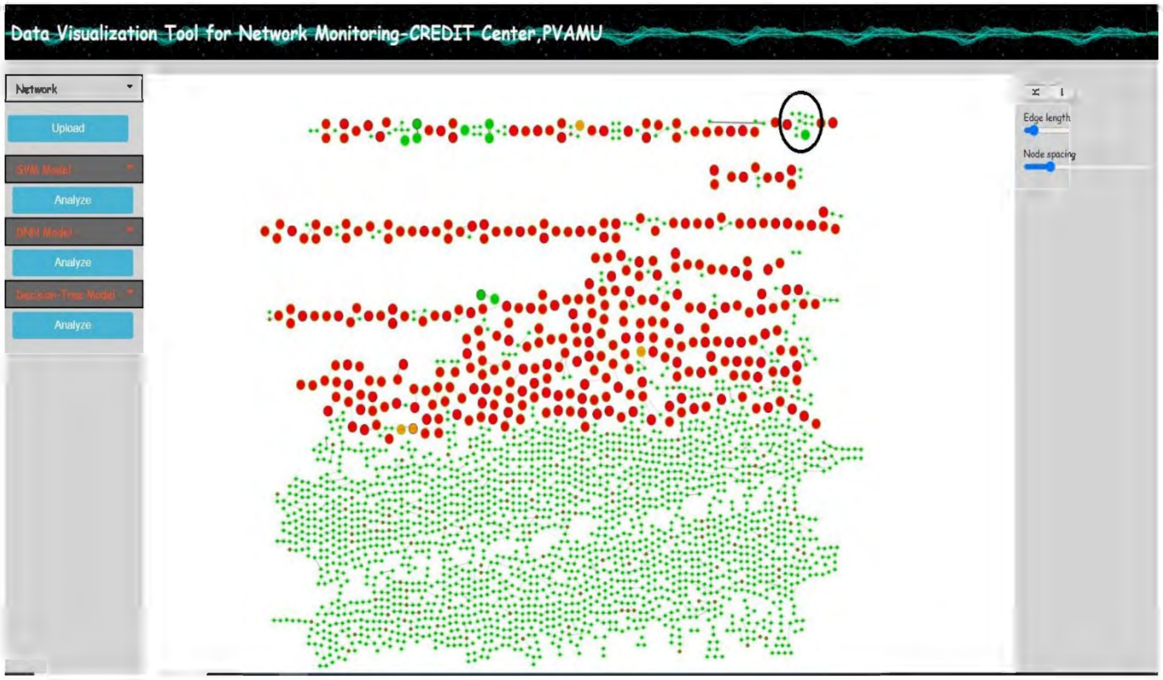


Fig. 4.10 Decision Tree Model output on Network Topology.

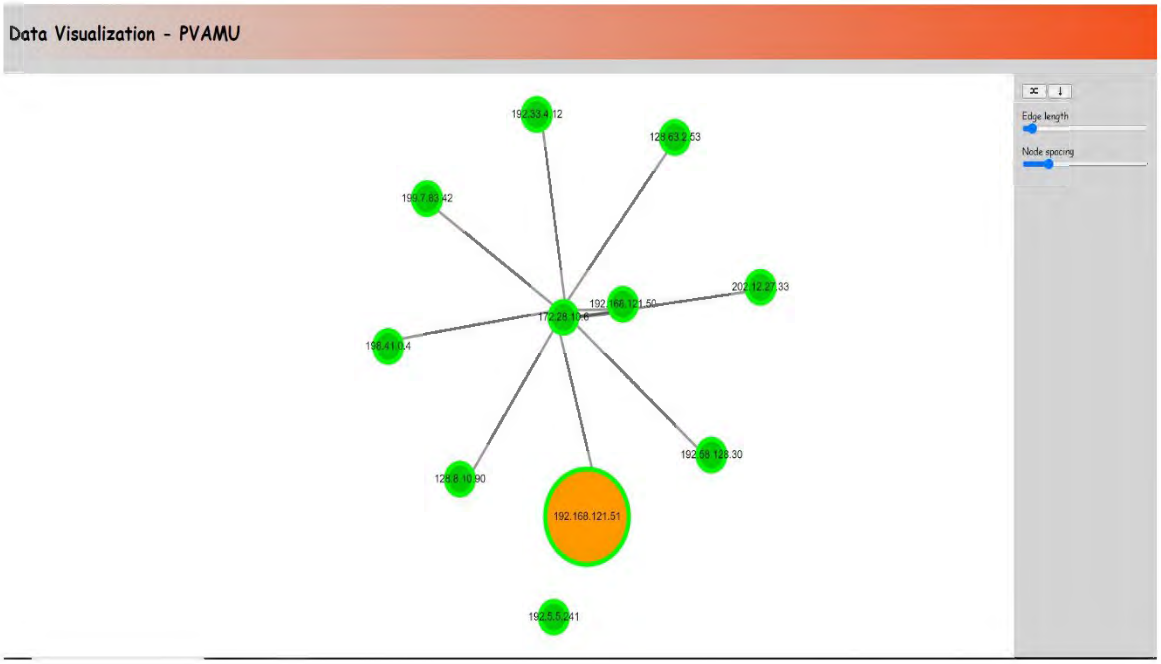


Fig. 4.11 Layered Network Topology.

Based on the results, can conclude that the research aims to design models like Decision Tree, SVM, and DNN at the backend was achieved. Similarly, designing visualization tools at the frontend and integrating them to successfully showcase the predicted attacks for network monitoring with layered network topology was also accomplished. Chapter 5 concludes and states the scope for future work.

V. CONCLUSION AND FUTURE WORK

Conclusion

This study aimed to build Deep Learning enhanced visualization tool for networking monitoring, which was achieved successfully. This work was accomplished in three phases.

The first goal was to build three Deep Learning models for predicting possible network attacks. Three models viz. SVM, Decision Tree, and DNN were designed and implemented using a python programming language. These three models were trained, tested, and analyzed to predict the DDoS attack on the IP network, and the "DARPA_2009_DDoS_attack-20091105" network dataset was used successfully. These models showed promising results, however, their accuracy has scope for improvement. The implemented Deep Learning framework achieved the first goal.

The second goal of this venture was to develop and deploy a visualization tool for network monitoring to tackle possible cyber-attacks. This goal was achieved with the help of an open-source platform visualization tool called Cytoscape. This tool is based on java-script libraries. Cola plugin libraries are used to get the desired graphical representation suited to the IP network topology. HTML, CSS, and JSON frameworks have been used to host the website for visualization tools in internet explorer. The goal of developing a visualization tool was accomplished.

The third goal was to integrate the Deep Learning models with the visualization tool to display the predicted output on the graphical user interface. Deep Learning models were programmed using the python language. The concern with the visualization tool was that the required input to the visualization tool was in JSON format, so REST API (application programming interface) was used to integrate the backend and frontend to build the complete Deep Learning visualization tool.

This Deep Learning enhanced visualization tool leveraged both the functionalities, predictions of possible cyber-attack, and representation of attack in the visualization tool as aimed.

Future Work

The Deep Learning enhanced visualization tool is robust and performs well predicting attacks. However, there is still a scope to improve the model performance by applying various techniques. This whole framework is flexible and loosely coupled. It could explore many possible options to change the backends' Deep Learning models or can change the frontend visualization tool with minimal chances in REST API. The model selection was made considering all sizes of datasets, from small to large. Still, it has limitations because of the power graphic card to display the number of nodes in the window. Accordingly, more work is expected on the scalability of the visualization tool to represent the nodes.

Given the expansive architecture of the IP network, this research aimed to optimize the visualization tool's multi-layered structure, effectively incorporating supplementary plugins. Nevertheless, Cytoscape libraries are not offering multi-layer structure format to show the subnetworks in layered format. Exploring Layered structures could help to get optimal performance from the visualization tool.

As an outcome of the above findings, the further step of this study would expand the prediction work to accomplish the high-performance models. Also, the study would aim to broaden the layering feature in Cytoscape libraries and explore the options on high-performance graphic cards to improve the scalability of the visualization tool. In addition, predictions on live incoming IP traffic would be explored to achieve affordable intrusion detection systems for small-scale companies.

REFERENCES

[1] Hoque, N., Bhuyan, M. H., Baishya, R. C., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network attacks: Taxonomy, tools and systems. *Journal of Network and Computer Applications*, *40*, 307-324

[2] Raiyn, Jamal. "A survey of cyber-attack detection strategies." *International Journal of Security and Its Applications* 8.1 (2014): 247-256.

[3] Cyber-security: Protecting Our Federal Government from Cyber Attacks, the 2009 data breach investigations report, 2009.

[4] Uma, M. and Padmavathi, G. (2013) A Survey on Various Cyber Attacks and Their Classification. International Journal of Network Security, 15, 390-396.

[5] R. Sandhu, Cyber-security: What You Need to Know Institute for Cyber-security (ICS), Oct. 2009.

[6] T. Shimeall, Cyberterrorism, Software Engineering Institution Carnegie Mellon University Pittspurg, pp 1-18, 2002.

[7] S. Cheung, “Modeling multistep cyber-attacks for scenario recognition,” in Proceedings of the Third DARPA Information Survivability Conference and Exposition, vol. I, pp. 284-292, Washington, D. C., Apr. 22-24, 2003.

[8] B. Cashell, W. D. Jackson, M. Jickling, and B. Webel, The Economic Impact of Cyber-Attacks, CRS Report for Congress, 2004.

[9] Barber R. Hacking techniques: the tools that hackers use and how they are evolving to become more sophisticated. Computer Fraud and Security 2001;2001(3):9–12.

[10] Lau, Felix, Stuart H. Rubin, Michael H. Smith, and Ljiljana Trajkovic. "Distributed denial of service attacks." In *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*, vol. 3, pp. 2275-2280. IEEE, 2000.

[11] Gu, Q. and Liu, P., 2007. Denial of service attacks. *Handbook of Computer Networks: Distributed Networks, Network Planning, Control, Management, and New Trends and Applications*, *3*, pp.454-468.

[12] Dittrich, D. (1999). The DoS Project's "trinoo" distributed denial of service attack tool. Available at: http://staff.washington.edu/dittrich/misc/trinoo.analysis. (Date of access: January 2, 2006)

[13] Handley, M. and Rescorla, E. (2006). Internet Denial of Service Considerations. Available at: http://tools.ietf.org/html/draft-iab-dos-05. (Date of access: October 31, 2006)

[14] Mirkovic, J., Dietrich, S., Dittrich, D., Reiher, P. (2005). Internet Denial of Service: Attack and Defense Mechanisms. Prentice Hall.

[15] CERT (1998). CERT Advisory CA-1998-01 Smurf IP denial-of-service attacks. Available at: http://www.cert.org/advisories/CA-1998-01.html. (Date of access: January 2, 2006)

[16] Kuzmanovic, A., and Knightly, E. W. (2003). Low-rate TCP-targeted denial of service attacks, Proceedings of ACM SIGCOMM, 75-86. ACM Press, New York

[17] CERT Advisory CA-1996-21 TCP SYN flooding and IP spoofing attacks. Available at: http://www.cert.org/advisories/CA-1996-21.html. (Date of access: January 2, 2006)

[18] https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/ntw2.12009

[19] Kaur, P., Kumar, M., & Bhandari, A. (2017). A review of detection approaches for distributed denial of service attacks. Systems Science & Control Engineering, 5(1), 301–320. doi:10.1080/21642583.2017.1331768

[20] Geetha, R. and Thilagam, T., 2021. A review on the effectiveness of Machine Learning and Deep Learning algorithms for cyber-security. *Archives of Computational Methods in Engineering*, *28*(4), pp.2861-2879.

[21] Tereykovskaya L.A. and Tereykovskiy I.A. Using the expertise in the development of neural network model for recognition of phonemes in the voice signal [Text] the proceedings of the II International scientific-practical conference Information and telecommunication technologies: education, science and practice, Almaty, Kazakhstan, 2015, pp. 258–261.

[22] Yemel'yanova Yu.G. Analiz problem i perspektivy sozdaniya intellektual'noy sistemy obnaruzheniya i predotvrashcheniya setevykh atak

na oblachnyye vychisleniya. [Analysis of issues and possibilities of development of smart system of detection and prevention of network attacks on cloud computations]. Program systems: Theory and application, 2011; 4: 17-31.

[23] Smith, R.G. and Eckroth, J., 2017. Building AI applications: Yesterday, today, and tomorrow. *AI Magazine*, *38*(1), pp.6-22.

[24] P. Louridas and C. Ebert, "Machine Learning", *IEEE Softw.*, vol. 33, no. 5, pp. 110-115, Sep./Oct. 2016.

[25] Jordan, M.I. and Mitchell, T.M., 2015. Machine Learning: Trends, perspectives, and prospects. *Science*, *349*(6245), pp.255-260.

[26] Sarker, I.H., Kayes, A.S.M., Badsha, S. *et al.* Cyber-security data science: an overview from Machine Learning perspective. *J Big Data* **7,** 41 (2020). https://doi.org/10.1186/s40537-020-00318-5

[27] Y. LeCun, Y. Bengio and G. Hinton, "Deep Learning", *Nature*, vol. 521, pp. 436-444, May 2015.

[28] L. Deng and D. Yu, "Deep Learning: Methods and applications", *Found. Trends Signal Process.*, vol. 7, no. 3, pp. 197-387, Jun. 2014.

[29] P. -W. Tsai, C. -W. Tsai, C. -W. Hsu and C. -S. Yang, "Network Monitoring in Software-Defined Networking: A Review," in *IEEE Systems Journal*, vol. 12, no. 4, pp. 3958-3969, Dec. 2018, doi: 10.1109/JSYST.2018.2798060.

[30] Ten, D.W.H., Manickam, S., Ramadass, S. and Al Bazar, H.A., 2009, November. Study on advanced visualization tools in network monitoring platform. In *2009 Third UKSim European Symposium on Computer Modeling and Simulation* (pp. 445-449). IEEE.

[31] Hideshima, Y. and Koike, H., 2006, January. STARMINE: A visualization system for cyber attacks. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation-Volume 60* (pp. 131-138).

[32] R. Sathya and A. Abraham, “Comparison of supervised and unsupervised learning algorithms for pattern classification,” International Journal of Advanced Research in Artificial Intelligence, vol. 2, no. 2, pp. 34–38, 2013.

[33]“A brief introduction to supervised learning.” https://towardsdatascience.com/. [Retrieved: September, 2019].

[34] V. Cherkassky and F. Mulier. Learning from Data - Concepts, Theory and Methods. John Wiley & Sons, New York, 1998.

[35] V. N. Vapnik. The Nature of Statistical Learning Theory. Springer, New York second edition, 2000.

[36] O. L. Mangasarian. Generalized Support Vector Machine. In A. Smola, P. Bartlett, B. Scholkopf, and D. Schuurmans, editors, Advances in Large Margin Classifiers, pages 135-146, Cambridge, MA, 2000. MIT Press. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-14.ps

[37] https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/

[38] https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[39] https://programmathically.com/understanding-hinge-loss-and-the-svm-cost-function/

[40] Patel, H.H. and Prajapati, P., 2018. Study and analysis of Decision Tree based classification algorithms. International Journal of Computer Sciences and Engineering, 6(10), pp.74-78.

[41] Gershman A, Meisels A, Lüke KH, Rokach L, Schclar A, Sturm A. A Decision Tree Based Recommender System. InIICS 2010 Jun 3 (pp. 170-179).

[42] Jadhav SD, Channe HP. Efficient recommendation system using Decision Tree classifier and collaborative filtering. Int. Res. J. Eng. Technol. 2016;3:2113-8.

[43] Song, Y.Y. and Ying, L.U., 2015. Decision Tree methods: applications for classification and prediction. Shanghai archives of psychiatry, 27(2), p.130.

[44] Charbuty, B. and Abdulazeez, A., 2021. Classification based on Decision Tree algorithm for Machine Learning. Journal of Applied Science and Technology Trends, 2(01), pp.20-28.

[45] RekhaMolala, “Entropy, Information Gainand Gini Index; the crux of a Decision Tree ,” Medium, Mar. 23, 2020. https://blog.clairvoyantsoft.com/entropy-information-gain-and-giniindex-the-crux-of-a-decision-tree-99d0cdc699f4 (accessed Dec. 28, 2020).

[46] V. Cheushev, D. A. Simovici, V. Shmerko, and S. Yanushkevich, “Functional entropy and Decision Tree ,” in Proceedings. 1998 28th IEEE International Symposium on Multiple-Valued Logic (Cat. No. 98CB36138), 1998, pp. 257–262

[47] Jojo John Moolayil,” A Layman’s Guide to Deep Neural Networks”, Medium, July. 24,2019. https://towardsdatascience.com/a-laymans-guide-to-deep-neural-networks-ddcea24847fb

[48] Montufar, G.F., Pascanu, R., Cho, K. and Bengio, Y., 2014. On the number of linear regions of deep neural networks. *Advances in neural information processing systems*, *27*.

[49] Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In NIPS, 2012.

[50] Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. In Proc. 30th International Conference on Machine Learning, pages 1319–1327, 2013.

[51] Ciresan, U. Meier, J. Masci, and J. Schmidhuber. Multi column deep neural network for traffic sign classification. Neural Networks, 32:333–338, 2012.

[52] Hinton, L. Deng, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition. IEEE Signal Processing Magazine, 29(6):82–97, Nov. 2012.

[53] Sagar Sharma,” Activation Functions in Neural Networks”, Medium, Sep 6,2017.https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

[54] Haoxiang, W. and Smys, S., 2021. Overview of configuring adaptive activation functions for deep neural networks-a comparative study. Journal of Ubiquitous Computing and Communication Technologies (UCCT), 3(01), pp.10-22.

[55] Ferrag, M.A., Maglaras, L., Moschoyiannis, S. and Janicke, H., 2020. Deep Learning for cyber-security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications*, *50*, p.102419.

[56] Kumari, Kimmi, and M. Mrunalini. "Detecting Denial of Service attacks using Machine Learning algorithms." *Journal of Big Data* 9, no. 1 (2022): 1-17.

[57] Mahajan, N., Chauhan, A., Kumar, H., Kaushal, S., & Sangaiah, A. K. (2022). A Deep Learning Approach to Detection and Mitigation of Distributed Denial of Service Attacks in High Availability Intelligent Transport Systems. *Mobile Networks and Applications*, 1-21.

[58] Brandon Williams.,2020.Intrusion Detection and Visualization Tool for Network Monitoring using Machine Learning.

[59] Amma, N.G. and Selvakumar, S., 2022. Optimization of vector convolutional deep neural network using binary real cumulative incarnation for detection of distributed denial of service attacks. Neural Computing and Applications, 34(4), pp.2869-2882.

[60] Orebaugh, A., Ramirez, G. and Beale, J., 2006. *Wireshark & Ethereal network protocol analyzer toolkit*. Elsevier.

[61] Deri, L., Martinelli, M. and Cardigliano, A., 2014. Realtime {High-Speed} Network Traffic Monitoring Using ntopng. In *28th large installation system administration conference (LISA14)* (pp. 78-88).

[62] Pras, A., Sperotto, A., Moura, G.C., Drago, I., Barbosa, R., Sadre, R., Schmidt, R. and Hofstede, R., 2010. Attacks by “Anonymous” WikiLeaks proponents not anonymous. *Design and Analysis of Communication Systems Group (DACS) CTIT Technical Report*, pp.1-10.

[63] Mansfield-Devine, S., 2011. Anonymous: serious threat or mere annoyance?. *Network Security*, *2011*(1), pp.4-10.

[64] Saxena, H. and Richariya, V., 2014. Intrusion detection in KDD99 dataset using SVM-PSO and feature reduction with information gain. *International Journal of Computer Applications*, *98*(6).

[65] R.Durst, T.champion, B.witten, E.Miller, and L.Spagnuolo, "Testing and valuating computer intrusion detection system" communications of ACM, Vo1.42, no.7, pp 53-61, 1999.

[66] Khan, L., Awad, M. and Thuraisingham, B., 2007. A new intrusion detection system using Support Vector Machine and hierarchical clustering. *The VLDB journal*, *16*(4), pp.507-521.

[67] Kim, D.S., Nguyen, H.N. and Park, J.S., 2005, March. Genetic algorithm to improve SVM based network intrusion detection system. In *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)* (Vol. 2, pp. 155-158). IEEE.

[68] Bhardwaj, A., Mangat, V., Vig, R., Halder, S. and Conti, M., 2021. Distributed denial of service attacks in cloud: State-of-the-art of scientific and commercial solutions. *Computer Science Review*, *39*, p.100332.

[69] Yassin W, Udzir NI, Muda Z, Sulaiman MN et al (2013) Anomaly-based intrusion detection through k-means clustering and Naives Bayes classification. In: Proceedings of the 4th International Conference on Computer Informatics ICOCI

[70] Saied A, Overill RE, Radzik T (2016) Detection of known and unknown DDoS attacks using artificial neural networks. Neurocomputing 172:385–393

[71] Tan Z, Jamdagni A, He X, Nanda P, Liu RP, Hu J (2014) Detection of denial-of-service attacks based on computer vision techniques. IEEE Trans Comput 64(9):2519–2533

[72] Liang, X. and Znati, T., 2019. On the performance of intelligent techniques for intensive and stealthy DDos detection. *Computer Networks*, *164*, p.106906.

[73] J. Mirkovic, G. Prier, and P. Reiher, "Attacking DDoS at the source," in International Conference on Network Protocols. IEEE, 2002, pp. 312–321

[74] Feinstein, L., Schnackenberg, D., Balupari, R. and Kindred, D., 2003, April. Statistical approaches to DDoS attack detection and response. In *Proceedings DARPA information survivability conference and exposition* (Vol. 1, pp. 303-314). IEEE.

[75] Xin, Y., Kong, L., Liu, Z., Chen, Y., Li, Y., Zhu, H., Gao, M., Hou, H. and Wang, C., 2018. Machine Learning and Deep Learning methods for cyber-security. *Ieee access*, *6*, pp.35365-35381.

[76] G. E. Hinton, ''Deep belief networks,'' Scholarpedia, vol. 4, no. 5, p. 5947, 2009.

[77] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, ‘‘Gradient-based learning applied to document recognition,’’ Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[78] Hideshima, Y. and Koike, H., 2006, January. STARMINE: A visualization system for cyber attacks. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation-Volume 60* (pp. 131-138).

[79] https://docs.microsoft.com/en-us/previous-versions/iis/6.0-sdk/ms525070(v=vs.90)

[80] https://docs.microsoft.com/en-us/iis/configuration/

[81] https://developer.mozilla.org/en-US/docs/Web/HTML

[82] https://www.w3schools.com/html/default.asp

[83] Yorgos Fountis,” How the Web works, Part II: What happens when you visit a URL”. https://pressidium.com/blog/how-the-web-works-what-happens-when-you-visit-a-url/ “

[84] https://developer.mozilla.org/en-US/docs/Web/CSS

[85] https://www.w3schools.com/cssref/

[86] https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference

[87] “*Cytoscape.js: a graph theory library for visualisation and analysis” ,*Franz M, Lopes CT, Huck G, Dong Y, Sumer O, Bader GD

[88] https://docs.python.org/3/reference/

[89] https://www.section.io/engineering-education/choosing-between-django-flask-and-fastapi/, James Sandy

[90] https://apiflask.com/

[91] X. Dong, U. Victor, and L. Qian, “Two-path deep semi-supervised learning for timely fake news detection,” arXiv, vol. abs/2002.00763, 2020.

CURRICULUM VITAE

Vaishali Dhemare **vaishali_dhemare@hotmail.com**

Objective:

A passionate and self-motivated student pursuing a master’s degree in Electrical Engineering have progressed through serial of roles over seven years from network engineer to systems engineer aimed to accomplish companies’ goals in line with its vision, mission, and corporate values. Possess extraordinary leadership qualities and strategic decision-making, with great vision and communication expertise.

Education:

PVPIT, Shivaji University, Maharashtra, INDIA
B.S. Electronics’ and Telecommunications Engineering Graduate GPA: 3.2
Graduation: May 2007

Prairie View A&M University, United states
M.S. Electrical Engineering Graduate GPA: 4.00
Graduation: Fall 2022

Employments:

Software Engineer, CSE ICON INC, United states Jan 2022 – Present
Software Engineering Intern, CSE ICON INC, United States, Sept 2021 – Dec 2021
Graduate Research Assistant, Center of Excellence in Research and Education for Big military Data Intelligence (CREDIT CENTER) united states, Aug 2019 – Sept 2021
Voice Switch Network Executive, Tata Telecommunications Pvt Ltd, India, Jun 2010 - Oct 2013
NGN Switch Engineer, ZTE Telecommunications Pvt Ltd, India, Apr 2008 – Jun 2010

Honors/Activities:

”Chevron Masters Student Scholership Award” Fall 2020 and Spring 2021