Attack Detection for Cyber Systems and

Probabilistic State Estimation in Partially Observable Cyber Environments

by

Sayantan Guha

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved June 2016 by the
Graduate Supervisory Committee:

Stephen S. Yau, Chair
Gail-Joon Ahn
Dijiang Huang

ARIZONA STATE UNIVERSITY

August 2016

ABSTRACT

Detecting cyber-attacks in cyber systems is essential for protecting cyber infrastructures from cyber-attacks. It is very difficult to detect cyber-attacks in cyber systems due to their high complexity. The accuracy of the attack detection in the cyber systems depends heavily on the completeness of the collected sensor information. In this thesis, two approaches are presented: one to detecting attacks in completely observable cyber systems, and the other to estimating types of states in partially observable cyber systems for attack detection in cyber systems. These two approaches are illustrated using three large data sets of network traffic because the packet-level information of the network traffic data provides details about the cyber systems.

The approach to attack detection in cyber systems is based on a multimodal artificial neural network (MANN) using the collected network traffic data from completely observable cyber systems for training and testing. Since the training of MANN is computationally intensive, to reduce the computational overhead, an efficient feature selection algorithm using the genetic algorithm is developed and incorporated in this approach.

In order to detect attacks in cyber systems in partially observable environments, an approach to estimating the types of states in partially observable cyber systems, which is the first phase of attack detection in cyber systems in partially observable environments, is presented. The types of states of such cyber systems are useful to detecting cyber-attacks in such cyber systems. This approach involves the use of a convolutional neural network (CNN), and unsupervised learning with elbow method and k-means clustering algorithm.

DEDICATION

This thesis is dedicated to the memory of my late father, Utpalendu Guha for being my constant source of inspiration. I am thankful for his grace and blessings.

ACKNOWLEDGEMENTS

I would like to thank Professor Stephen S. Yau for providing me motivation, guidance and the opportunity to work on this research project. He is a source of inspiration to me. I would also like to thank my committee members Professors Gail-Joon Ahn and Dijiang Huang for being the part of my graduate supervisory committee.

The invaluable inputs I have received for my thesis from Arun Balaji Buduru and other members of Professor Yau's research laboratory in the Information Assurance Center are very much appreciated. I wish my gratitude to all of my teachers of South Point High School and Heritage Institute of Technology for encouraging to study further at graduate school.

I would like to thank my mom, Dolly Guha for supporting and inspiring me for never giving up on trying and teaching me that you can achieve good results with hard work. Most importantly, I am grateful to my wife, Priyanka for her support, encouragement and patience and taking care of me during hard times.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

Chapter 1

INTRODUCTION

Security is the most serious concern of many applications of cyber systems, and detecting cyber-attacks (referred as “attacks” in this thesis) on cyber systems is important for protecting cyber systems. It is difficult to detect attacks to cyber systems due to the complex natures of the cyber systems [1]. In addition, the provision of access from various mobile smart and stationary devices to cyber systems increases the difficulty and complexity of attack detection. The accuracy of the attack detection in the cyber systems depends heavily on the completeness of the collected sensor information. The information collected from the sensors may be from completely observable or partially observable.

The types of states exhibit much unusual variations of features of network traffic when attacks occur on cyber systems, and hence are useful to security [2] for detecting cyber-attacks and predicting security breaches [3] using Bayesian Network [4] and Markov Decision Process [5]. However, estimating the types of states of cyber systems in partially observable environments is much more difficult because certain features of the input data may be missing.

In this thesis, effective approaches are presented to address these two problems:

1. Detecting attacks in completely observable cyber systems.
2. Estimating types of states in cyber systems in partially observable environments.

Existing attack detection techniques for cyber systems have the following serious limitations. The signature-based attack detection techniques [6, 7] have low detection rate for zero-day attacks. Anomaly-based attack detection techniques for cyber systems

[8] are difficult to use because monitoring and analysis of a large amount of network and system events in cyber systems simultaneously are difficult and time-consuming. The specification-based attack detection [9] is difficult to implement as it is very difficult to define all possible legal specifications and it is also computationally intensive [10].

Machine learning may be used in attack detection methods in the mentioned three techniques, but they usually are computation intensive [11, 12]. In this thesis, we will present an approach to detecting attacks on cyber systems, including access from remote computing devices using a multimodal artificial neural network (MANN) [13] and a genetic algorithm [14] to reduce computational overhead. We will illustrate our approach using comprehensive network traffic data sets of cyber systems to show that our approach will generate better results than existing techniques [11, 12, 8, 15, 16, 17].

The existing techniques to estimation of types of states of the cyber system also have the following serious limitations. Hartikainen et al. [18] presented the filter-based technique for the detecting changes of the network states. However, this technique needs the complete information of the system. In [19], the state estimation of the interconnected networks was performed for network security analysis using a stochastic Extended Kalman Filter (EKF) algorithm. However, the Extended Kalman Filter algorithm are prone to failure when the system is non-linear. An approach with predictive capability was proposed in [3] to predict system-wide security breaches in cyber systems using Bayesian Network (BN) [4] and Markov Decision Process (MDP) [5]. However, this technique requires inputs from the domain experts for the estimation of the initial states, based on the system state dependencies of the cyber systems. Silver and Veness [20] presented an approach to estimating types of states using partially observable Markov decision process (POMDP), where the agents take deterministic actions in a partially observable environment.

Hence, for first problem, we will present an approach to detecting attacks of completely observable cyber systems, which include access from remote computing devices. Two comprehensive network traffic data sets [16, 17]will be used for training and testing our approach, and we will show that the performance of our approach is better than the existing approaches [15, 17, 21, 22].

For the second problem, in order to detect attacks in cyber systems in partially observable environments, an approach to estimating the types of states in partially observable cyber systems, which is the first phase of attack detection in cyber systems in partially observable environments, is presented. A comprehensive network traffic data sets [23] will be used for training and testing our approach to estimating types of states in partially observable environments, and we will show that the performance of our approach is better than the existing machine learning techniques [24, 25, 26].

The thesis work is organized as follows: Chapter 2 describes the current state of the art for attack detection in cyber systems and estimating types of states. In Chapter 3, the overall approach for detecting cyber-attacks in completely observable cyber systems is presented. In Chapter 4, we present the details of detecting cyber-attacks in completely observable cyber systems. In Chapter 5, we will illustrate our approach using two examples with two comprehensive network traffic data sets and compare the performance of our approach with existing approaches. In Chapter 6, we discuss an overall approach for estimating the types of states in partially observable environments. In Chapter 7, we present the detailed techniques to estimate the types of states of the cyber systems in partially observable environments. In Chapter 8, we illustrate our approach of estimating the types of states of the cyber systems in the partially observable environments. Finally, we conclude the thesis discussion in Chapter 9 with future work and conclusion.

Chapter 2

CURRENT STATE OF ART

2.1 Detecting Attacks in Cyber Systems

Existing attack detection techniques for cyber systems have the following serious limitations. The signature-based attack detection techniques [6, 7] have low detection rate for zero-day attacks. Anomaly-based attack detection techniques for cyber systems [8] are difficult to use because monitoring and analysis of a large amount of network and system events in cyber infrastructures simultaneously are difficult and time-consuming. For machine-learning type attack detection, Vieira et al. [11] presented a technique using an artificial neural network (ANN) for cyber systems, but this technique requires a large amount of time to detect attacks and was based on the assumption that all the events of the cyber systems are predictable. Su et al. [12] presented a fuzzy logic based attack detection technique, but the technique takes much time for training the attack detection algorithm. Modi, et al, [8] presented an attack-detection approach based on associations among features of network traffic in cyber infrastructures and support vector machine, but the technique can only be used for classifying discrete features of the network traffic.

2.2 Estimating Types of States in Partial Observable Cyber Environments

Most of the existing approaches to estimate types of states are based on applying filter-based method. Erik, et al [18], presented filter-based method for network state estimation in to detect a change in the network systems. However, this technique needs complete information of the network systems. In [19], the state estimation of the interconnected network was performed for network security analysis using stochastic Extended Kalman Filter (EKF) algorithm. However, the Extended Kalman Filter are prone to failure when the system is non-linear and assumes that the state belief is Gaussian distributed. EKF assumes that both observation models and system equation are non-linear which is unrealistic for practical situations.

In [27], Moshe, et al, proposed a system to estimate the current state of the system and predict future conditions for traffic congestion problem. However, the proposed system is based on the demand and supply stimulation. In practical application, this approach has the difficulty of rationality and the approach does not consider the partially observable environment.

Estimating states for predicting potential security breaches on cyber infrastructures was proposed in [3] through system-wide causal relationship and probabilistic human behavior. However, this technique requires input from the domain experts and developing an effective data specification language for domain experts to provide their input is more complicated. Moreover, the approach also assumes that the initial system state dependencies are known to the domain expert.

The estimation of states can be done using a partially observable Markov decision process (POMDP) - a generalization of Markov decision process (MDP) [5], where the agents take deterministic actions in a dynamic and partially observable environment. However, POMDP has some serious drawbacks as generating universal plans for actions become computationally complex which is less useful in practical applications [20]. Another disadvantage of POMDP is that finding the optimal policy is inflexible and the existing solutions can handle no more than few hundred states [28]. Hence, POMDP cannot be applied for estimating the states of the cyber system as there may exist several thousand of states in a cyber system.

In [29], the technique for estimating states for the traffic congestion control problem was proposed using a hidden Markov model (HMM). However, HMM can be applied where there are discrete “hidden” states and it cannot be applied for a continuous variable. Moreover, HMM is expensive both in terms of memory and compute time as it is difficult to compute HMM with a large number of states.

Chapter 3

OVERALL APPROACH TO DETECTING ATTACKS IN COMPLETELY OBSERVABLE CYBER SYSTEMS

In this chapter, we will present our overall approach for detecting attacks on completely observable cyber systems. We assume that the cyber systems have been used for a period of time, and substantial traffic data on the cyber systems, including the traffic data with attacks, has been collected during this period. The attacks detected in our approach are of known categories and all the sensors (monitors) are reliable and always work. The features extracted from the network traffic data [30] contain information, such as source packet size, destination packet size, protocol types, flag and transmission duration, which are sufficient for detecting attacks in cyber systems [16]. In our approach, the labeled data sets of network traffic are used for training and testing of supervised machine learning in multimodal artificial neural network (MANN). The categories of attacks that can be detected depend on the available labels in the training data set.

Our approach uses the MANN [13] for classifying various categories of attacks on cyber systems because MANN is relatively simple to implement and generates better results than existing methods due to the fact that it identifies the nonlinear relationship between dependent and independent variables (features) in a data set [31]. Since the training of MANN involves heavy computation, we develop an effective feature selection algorithm for reducing the amount of computation for classifying attacks [32] using a genetic algorithm [14]. The MANN detects the attacks in the cyber systems by classifying monitored network traffic data into different attack categories. The MANN

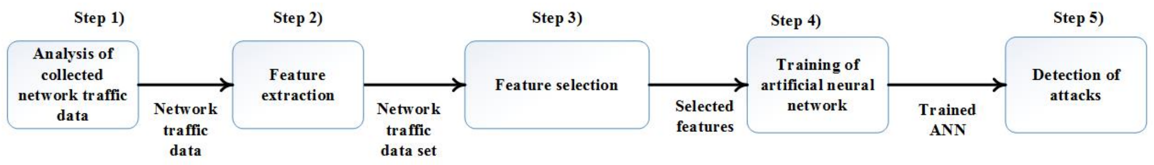


Figure 1: System Diagram of our approach

produces better classification results by minimizing the number of incorrectly classified cases during the training phase [31]. During the testing phase, the MANN analyses the input testing data set. The trained machine learning algorithm is implemented in the cyber systems as attack detection web service. The attack detection web service has the advantage to be used by remote computing devices, both mobile, and stationery, to detect attacks in cyber systems [33].

Our attack detection approach for cyber systems can be summarized as follows, and is depicted in Figure 1:

Step 1) Analyze the collected traffic data on each connection link of the network of the cyber systems using tools, such as packet sniffer tools [34] to determine the categories of past attacks to the cyber systems. Continue to collect the data and update the data.

Step 2) After collecting substantial network traffic data, extract the features of individual packets from the collected data using tools, such as tcptrace [35]. Generate the training data set and the testing data set by applying a cross-validation method [36].

Step 3) Select effective features for attack detection from the packets in the training and testing data generated in Step 2) using a genetic algorithm [14].

Step 4) Train the MANN using back-propagation algorithm [37] and sigmoid activation function [37] with the effective features selected for attack detection in Step 3).

Step 5) Use the trained MANN to detect attacks in the input network traffic data (unlabeled).

In our approach, the network traffic data set is used as network traffic is encapsulated in network packets and it provides various security parameters of cyber systems. In cyber systems, the network data is captured in the form of network packets. In Step 1), packet sniffer tools, such as TCPDUMP [34] and Wireshark [38], are used to capture network traffic and are used for intercepting and displaying network packets on the network interface. The output of these tools is the packet capture file (pcap).

In Step 2), each extracted feature contains the packet-level information on each connection link. The network traffic data is labeled based on the attack categories determined in Step 1). Cross validation is used to generate the data sets for training and testing as there is no overlapping between training and testing data sets [36]. In cross-validation method [36], the network traffic data is randomized into k equal size partitions. The k^{th} partition is selected as a testing data set and $(k-1)$ partitions are selected as the training data set. Since the training data set is labeled, we can estimate the classification accuracy of our trained MANN by comparing the estimated label and the actual label.

In Step 3), the effective features are selected from labeled network traffic data set using a genetic algorithm. The feature selection algorithm will select the feature subset from the overall set of features of network traffic data. We also use multimodal neural network (MNN) in feature selection to evaluate the accuracy. The reduced number of features will reduce the computational complexity of training of MANN.

In Step 4) , multimodal artificial neural network (MANN) is selected as a machine learning algorithm. The selected features identified in Step 3) are fed as inputs into MANN for training the machine learning algorithm. The trained MANN is used to detect attacks in the cyber systems based on the attack categories. The back-propagation algorithm and a sigmoid activation function is used to train the MANN. In Step 4), the trained MANN is implemented as attack detection service which can be used by various computing and storage devices and applications (both mobile and stationary) connected to cyber systems to detect the attack. The testing data are feed from remote computing devices into the trained MANN to test attack detection using feed-forward algorithm. The testing data which is used during the attack detection phase contains attacks along with the normal data. MANN takes each observation of network traffic from the testing data set and classifies them based on the attack categories.

In Step 5), the trained MANN is used to detect attacks in the cyber systems by feeding the input data of network traffic (unlabeled).

Thus, the thesis aims at techniques to select an effective features from the network traffic data set for the detection of attacks. The reduced number of features improve the overall efficiency for training of the MANN without significantly affecting the overall accuracy metrics. The thesis also provides an innovative approach for attack detection service which is used as Security-as-a-Service (SECaaS) [39] for cyber systems. In this thesis, we aim to detect attacks of known attack categories. Our trained ML algorithm will aim to detect those attack categories which are mentioned in the training data set.

Chapter 4

DETECTING ATTACKS IN COMPLETELY OBSERVABLE CYBER SYSTEMS

4.1 Overview

This chapter explains the technique used to detect cyber-attacks in completely observable cyber systems. In Section 4.2, we present the technique of selecting features of the network traffic data set using a genetic algorithm. In Section 4.3, we present the technique to train the supervised machine learning algorithm using the multimodal artificial neural network (MANN). In Section 4.4, we present the approach of implementing the trained machine learning algorithm as web service in cyber systems. In Section 4.5, we present the technique to detect cyber-attacks in the cyber systems from remote computing devices.

4.2 Feature Selection

The features need to be selected carefully from the network traffic data set for training the MANN; else it will degrade the performance and accuracy to detect the cyber-attacks in the cyber systems. Researchers have proposed many feature selection algorithm, but they have comprised either efficiency or accuracy. Our feature selection algorithm maintains a trade-off between accuracy and efficiency for detecting attacks in the cyber systems.

The goal of our feature selection is to select a proper subset of the set of all features generated in Step 2) that reduces the computational complexity without

Algorithm 1 Feature Selection Algorithm using Genetic Algorithm

```
Input: Final Feature Set $F_{final} = \{\}$
Output: Final Feature Set
    Initialization : Initialize all Groups = $\{g_1, g_2, g_3, ...., g_n\}$.
    Calculate accuracy for each group using multimodal neural network classifier,
    Calculate overall accuracy by applying all features $\{f_z\}$ using multimodal neural
    network,
    Sort the Groups according to the accuracy of each group
1:  while accuracy of the set of the remaining features drops below the error margin
    of original overall accuracy or all the feature sets are exhausted do
2:      for each group $g_i \in \{g_1, g_2, ....., g_m\}$ do
3:          Pick the group with the lowest accuracy and eliminate the group $\{g_n\}$ from
            set of groups.
            Group Elimination Step
4:      end for
5:      for each feature of discarded group$\{g_n\}, f_a \in \{g_n\} = \{f_1, f_2, ....., f_b\}$ do
6:          Select a set of feature $\{f_x\}$ from $\{f_a\}$ such that $f_x \subseteq \{f_a\}$
            Feature Selection Step
            Feature selection is done using Genetic Search algorithm and multimodal
            neural network classifier
7:      end for
        Add the selected feature into the Final Feature Set $F_{final}$
        Calculate the overall accuracy by applying the remaining set of features $f_x \cup$
        $(f_z - f_a)$ using multimodal neural network
8:  end while
```

significantly affecting the overall accuracy metrics. In order to achieve this goal, we use a genetic algorithm for feature selection [14]. The genetic algorithm has the capability of effectively solving optimization problem and the ability to handle multiple solution search spaces [40]. Hence, the genetic algorithm can determine the effective features that affect the overall accuracy for attack detection. In order to estimate the effectiveness of the features, we use multimodal neural network (MANN) algorithm [41] to generate the performance metrics. As MANN has the ability to detect nonlinear relationship between the dependent and independent features, it can detect the possible interactions among features of the network traffic data.

In order to benchmark and assess the effectiveness of the feature selection technique, as presented in Algorithm 1, we calculate the accuracy of the MANN for the data set with all the features. Our feature selection technique then groups the features into feature sets based on the feature similarity in terms of characteristics of network packets, such as payload, transmission duration, and protocol. The feature sets are then evaluated individually with the MANN to determine their accuracy.

The genetic algorithm is applied to the individual feature sets starting with the one having the lowest accuracy. The genetic algorithm then identifies the irrelevant features based on the fitness function with the objective to improve the accuracy while reducing the number of features [14]. All the irrelevant features are eliminated from the overall feature set and the remaining features are evaluated with MANN and accuracy are generated. The process is repeated on the feature set with the next lowest accuracy until we exhaust all the feature sets or when the accuracy of the set of the remaining features drops below the error margin of original overall accuracy of the MANN model with all the features. The error margin can be set heuristically to overcome the local optimum problem. The acceptable error margin of accuracy for attack detection in cyber system is set by domain expert. The procedure for the feature selection based on genetic algorithm is shown in Figure 2.

4.3 Training of Supervised Machine Learning Algorithm

In this section, we explain how to use a multimodal artificial neural network (MANN) as a machine learning algorithm to detect attacks in cyber systems.

The input features selected in the previous section are used to train the machine

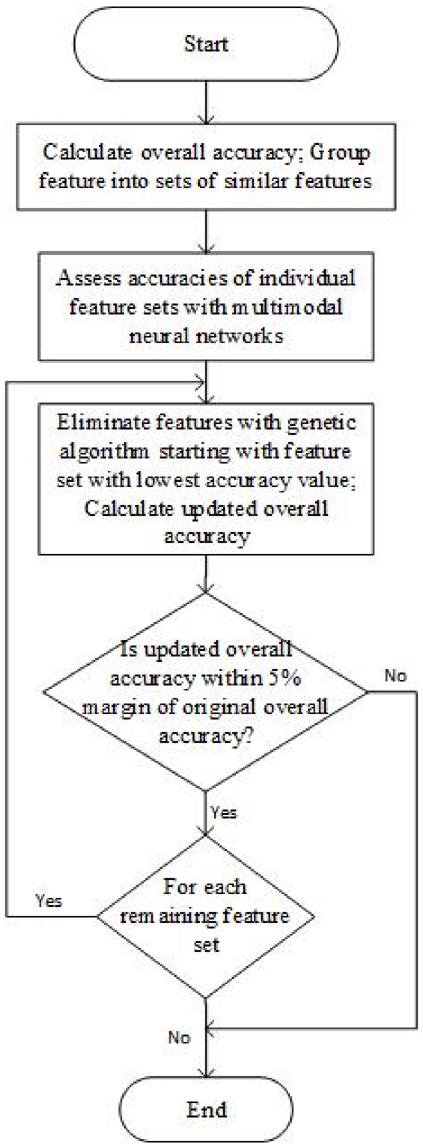

Start
Calculate overall accuracy; Group feature into sets of similar features
Assess accuracies of individual feature sets with multimodal neural networks
Eliminate features with genetic algorithm starting with feature set with lowest accuracy value; Calculate updated overall accuracy
Is updated overall accuracy within 5% margin of original overall accuracy?
No
Yes
For each remaining feature set
Yes
No
End

Figure 2: Our feature selection algorithm

learning algorithm. In this section, we use the MANN as machine learning algorithm which will help to determine attacks in the cyber systems.

4.3.1 Artificial Neural Network

An artificial neural network (ANN) is composed of many artificial neurons which are connected together is specific network architecture. The objective of the artificial neural network is to generate significant outputs from the provided inputs.

The ANN consists of three layers:

1. Input Layer: The features selected are fed into the ANN from the input layer.
2. Hidden Layer: The activities of the input layer and the weight on the connection between hidden and input layer determine the activity of the each hidden layer.
3. Output Layer: The activity of the output layer is determined by the activity of the hidden layer and the weights on the connection between hidden and an output layer.

For binary classification, the output layer of the ANN contains two nodes, whereas for the MANN the number of nodes in the ouput layer depends on the number of class labels. Multimodal artificial neural network (MANN) is used to learn representations from multiple modalities. The input layer of the MANN has the same number of nodes as the number of features selected determined in Step 3). The number of nodes in the output layer is the same as the number of attack categories for detection. Only one hidden layer is needed to overcome the vanishing gradient problem [42]. The sigmoid function is used as the activation function of the MANN as sigmoid function is bounded, monotonic, and differentiable [43]. The MANN is fully-connected, which means that the output layer of the MANN is fully connected to the hidden layer and

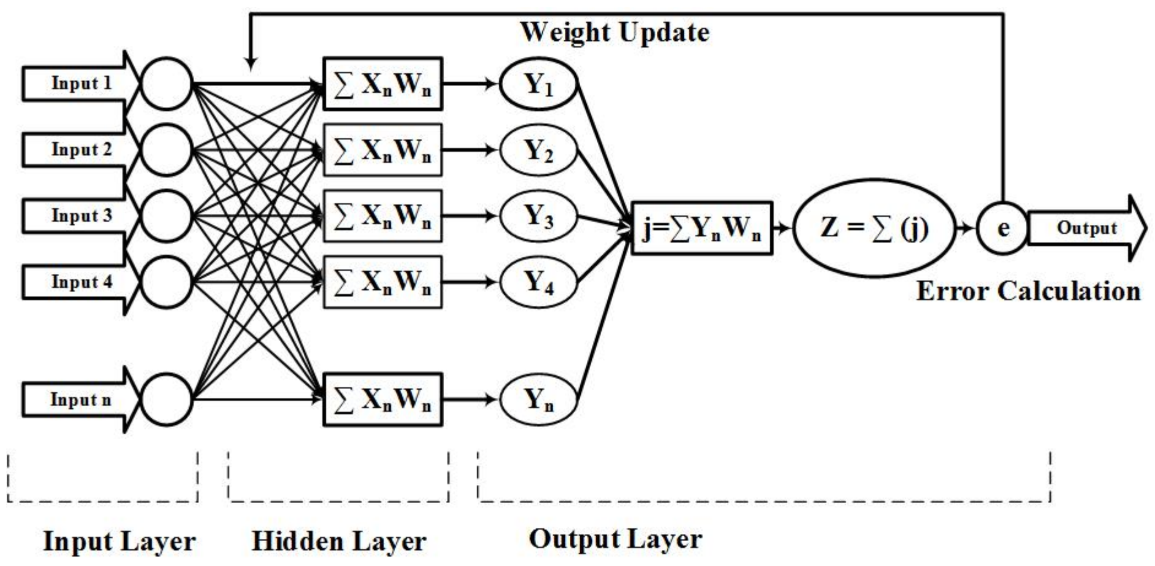


Figure 3: Multimodal artificial neural network with back-propagation algorithm

the hidden layer is fully connected to the input layer. The learning rate of the MANN is set to 0.1 since a higher rate can easily overshoot local minima. The learning iteration of the MANN is set 50 because higher learning rate will slow down the training process. The number of nodes in the hidden layer is taken as 22 since the number of hidden layer nodes is generally set approximately two-third of the number of nodes in the input layer [44].

4.3.2 Back Propagation Neural Network

In the training phase, the back-propagation algorithm [45] is used to train the MANN. The training data set is fed as input to MANN and the weights of input layer which are randomly assigned are multiplied with the input data and it goes into the hidden layer. The nodes of the MANN activate depending on the output of the sigmoid function. In back-propagation algorithm, the errors from output layer are back-propagated to the input layer. As the hidden layer does not have target values, the back-propagation algorithm helps to fine tune the weights. The calculated value is compared with the target value of the output layer and the connection weights are adjusted continuously as the errors back-propagated through the nodes. This weight updating process will continue till error, measured by comparing target values and real values stops improving. Figure 3 provide the diagram for an MANN with back-propagation algorithm. Algorithm 2 describes the back-propagation algorithm used in the training of MANN. The back-propagation algorithm [45] is used to train the MANN as follows:

Step 4.1) Randomly assign the initial weights of the edges of the input layer of the MANN, and use an arbitrary number for the initial threshold value of the activation function.

Step 4.2) Update the weights of the edges with backpropagation algorithm [45] using the training data set in Step 2).

Step 4.3) Repeat Steps 4.1) – 4.2) until the error rate between the correct output and the actual output stabilizes for each epoch.

Algorithm 2 Back Propagation Algorithm

```
Initialize weights(typically random)
repeat
  Keep doing epochs
  for all Example e in training set do
    Forward pass to compute
    $O \leftarrow neural - net - output(network, e)$
    $miss \leftarrow (T - O) ateachoutputunit$
    backward pass to calculate deltas to weights
    update all weights
  end for
until Tuning set error stops improving
```

The sigmoid function [46] used in the hidden layer. Let $g(x)$ represents the

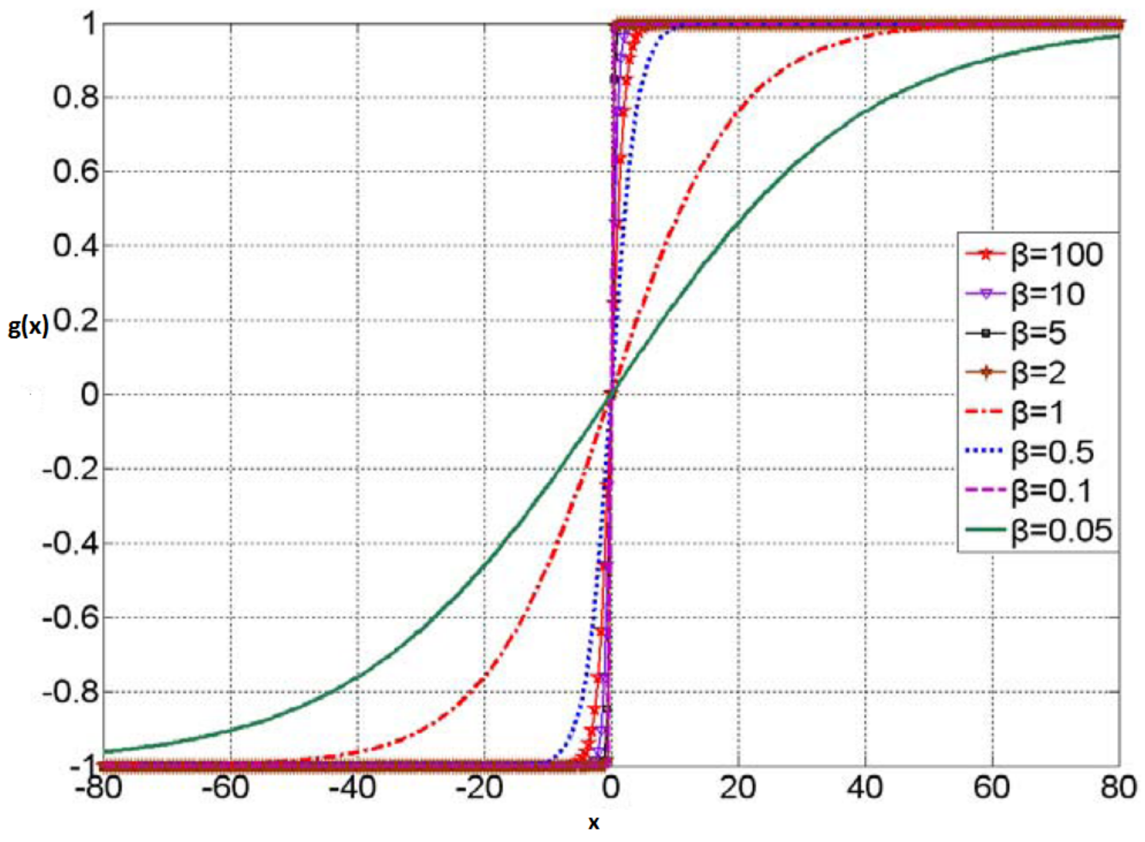


Figure 4: Sigmoid function

mathematical function of sigmoid function of input x and $\beta \in R^+$ [47] is the slope parameter. Figure 4 is the graph of the sigmoid function. Sigmoid function is special case of linear regression [48] and is defined by:

$$g(x) = 1/(1 + exp(-\beta x))$$

Here each node passes the output to the next layer and finally it reaches the output layer. The error 'e' is calculated by comparing the real output θ_R and the final output θ_F.

$$e = (\theta_R - \theta_F)/\theta_R$$

The error is used to update the weights in the neural network. Multimodal artificial Neural Network can deal with both discrete and continuous variables in the same way. The training of the MANN terminates when the value of e stops improving.

4.4 Implementing Trained Machine Learning Algorithm as Web Service

The multimodal artificial neural network is trained using training data set using back propagation algorithm. We implemented the trained MANN as web service in the completely observable cyber systems which can be used as attack detection service by the various applications connected to the completely observable cyber system to determine the attacks. The web service is exposed for external use to detect attacks in the completely observable cyber system. The trained attack detection algorithm can be used as Security-as-a-Service (SECaaS) [39] by various normal as well as resource constrained devices to detect an attack in completely observable cyber systems and thus protecting from constant security threats.

4.5 Detecting Attacks in Completely Observable Cyber System

The testing data set is fed into the input layer of the trained MANN. The feed-forward algorithm [13] is used in the MANN to classify network traffic data into different attack categories. The label of the classified data of the testing set and the actual label of the testing data set are compared to determine the accuracy for attack detection.

Chapter 5

CASE STUDY OF DETECTING ATTACKS IN COMPLETELY OBSERVABLE CYBER SYSTEMS

5.1 Overview

In this chapter, a case study for detecting attacks in the completely observable cyber systems is provided. In 5.2, the description of two comprehensive network traffic data sets is provided. In 5.3, the results of each step of our feature selection algorithm are provided. In 5.4, the steps to train of multimodal artificial neural network (MANN) are shown. In 5.5, we conclude with the results of the detection of attacks in the completely observable cyber systems.

5.2 Description of the Network Traffic Data Set

We use two comprehensive network traffic data sets, the NSL-KDD Cup [49] and UNSW-NB15 [16] data sets to train the MANN and detect attacks in the completely observable cyber systems. We use Microsoft Azure to implement and facilitate the training of supervised machine learning. The NSL-KDD data set is based on the original KDD'99 data set which is collected by DARPA as compressed raw TCP dump data for a period of 7 weeks. The dataset has approximately 4,900,000 single connection links and each contains 43 features. On the other hand, the UNSW-NB15 dataset was created by the Cyber Range Lab of the Australian Centre for Cyber

Security (ACCS). The data set contains 257,673 records of network traffic and has 49 features.

In the NSL-KDD Cup data set, there are the following four categories of attacks:

1. Denial of Service (DoS) : In this type of attacks, the attacker tries to consume resources to deny requests or authorized users to access the systems.
2. Probing Attack (Probe) : The attacker attempts to gain access security controls by gathering information about the systems.
3. Remote to Local Attack (R2L): In this attack, the attacker who is unauthorized to a system gains access to the system by sending packets over the network.
4. User to Root (U2R) : The attacker tries to exploit some vulnerabilities to gain root access to the system after gaining access to a normal user account on the system.

In UNSW-NB15 data set, there are nine categories of attacks:

1. Fuzzers: In this attack, randomly generated data is feed into a suspend program or network.
2. Reconnaissance: Attacker gathers information from the system and stimulates the attacks.
3. Shellcode: It is code used as the payload of a network packet to exploit network attacks.
4. Analysis: This attack includes port scan, spam and HTML files penetrations.
5. Backdoors: Access of a system is gained by silently bypassing the security mechanism.
6. Denial of Service

7. Exploits: The attacker exploits the vulnerabilities of the system through the known loopholes of the system.
8. Generic: The attack is implemented without knowing how the cryptographic primitive is implemented and works for all block ciphers.
9. Worms: The attack replicates itself to spread through the network.

As various services are provisioned through the internet, cyber systems suffer from the different types of attacks as mentioned in these two data sets. Hence, these two data sets are suitable for detecting attacks in completely observable cyber systems as these attack categories are common in cyber systems.

5.3 Attack Distribution

In NSL-KDD data set, few additional attack types are added testing data set which is not present in the training data set. Hence, the NSL-KDD data set is more robust to evaluate MANN to detect attacks in the completely observable cyber systems. Each attack category may contain different attack types. Table 1 and Table 2 exhibit the attack-type based on the attack category for NSL-KDD Cup [49] and UNSW-NB15 [16] data set respectively.

Step 1) and Step2) of our approach have already been done for the NSL-KDD Cup and UNSW-NB15 data sets. Hence, we will start with Step 3) i.e. with feature selection.

Table 1: Attack Types in NSL-KDD dataset

Attack Category	**Attack-type in NSL-KDD Training dataset**	**Additional attack-type in NSL-KDD Test dataset**
DoS	back, neptune, smurf, teardrop, land, pod.	apache2, mailbomb, processtable
Probe	satan, portsweep, ipsweep, nmap	mscan, saint
R2L	warezmaster, warezclient, ftpwrite, guesspassword, imap, multihop, phf, spy	sendmail, named, snmpgetattack, snmpguess, xlock, xsnoop, worm
U2R	rootkit, bufferoverflow, loadmodule, perl	httptunnel, ps, sqlattack, xterm

5.4 Feature Selection

In this section, we will select the number of features required to train multimodal artificial neural network using a genetic algorithm. Based on the identified five characteristics (fundamental, payload, time dependent, system dependent, and data dependent) of every feature of network traffic in the NSL-KDD data set, we have five groups, one group for one of these five characteristics. For UNSW-NB15 data set, each feature has the five characteristics (basic, content, time, general purpose, and connection) of features, and hence the data set is also divided into five groups. Table 3 and Table 4 details the distribution of features of NSL-KDD Cup and UNSW-NB15 data sets respectively according to the groups.

For our algorithm, we use multimodal neural network (MANN) classifier for accuracy measurement and performed genetic search using Weka. In the beginning, the accuracy of each group, as well as the overall accuracy (including all features),

Table 2: Attack Types in UNSW-NB15 dataset

Attack Category	**Attack-type in UNSW-NB15 Training and Testing data set**
Fuzzers	FTP, HTTP, RIP, SMB, Syslog, PPTP, DCERPC, OSPF, TFTP, DCERPC, BGP
Reconnaissance	Telnet, SNMP, SunRPC Portmapper, NetBIOS, DNS, HTTP, ICMP, SCTP, MSSQL, SMTP
Shellcode	FreeBSD, HP-UX, NetBSD, AIX, SCO Unix, Linux, Decoders, IRIX, OpenBSD, Mac OS X, BSD, Windows, BSDi, Multiple OS, Solaris
Analysis	HTML, Port Scanner, Spam
Backdoors	Backdoors
DoS	Ethernet, Microsoft Office, VNC, IRC, RDP, TCP, VNC, FTP, LDAP, Oracle , TCP, TFTP, DCERPC, XINETD, IRC, SNMP, ISAKMP, NTP, Telnet, CUPS, Hypervisor, ICMP, SunRPC, IMAP, Asterisk, Browser, Cisco Skinny, SIP, SMTP, SNMP, SSL, TFTP, SMTP, DNS, IIS Web Server, Miscellaneous, RTSP, Common Unix Print System (CUPS), SunRPC, IGMP, Microsoft Office, HTTP, LDAP, NetBIOS/SMB, Oracle, Windows Explorer
Exploits	Evasions, SCCP, SSL, VNC, Backup Appliance, Browser, Clientside Microsoft Office, Interbase, Miscellaneous Batch, SOCKS, TCP, Apache, IMAP, Microsoft IIS, SOCKS, Clientside, Microsoft Paint, IDS, SSH, ICMP, IDS, DCERPC , FTP, RADIUS, SSL, WINS, Clientside Microsoft, POP3, SSH, TCP, Unix Service, WINS, Cisco IOS, Clientside Microsoft Media Player, Dameware, IMAP, LPD, MSSQL, Office Document, RTSP, SCADA, VNC, Webserver, All, LDAP, NNTP, Office Document, RTSP, IGMP, Oracle, RDesktop, Telnet, Unix 'r' Service, LPD, All, Apache, ICMP , Microsoft IIS, PHP, SMB, SunRPC, Web Application, PHP, DNS, Evasions, NNTP, SMTP, RADIUS, Browser FTP, Miscellaneous, PPTP, SCCP, SIP, TFTP
Generic	SIP, HTTP, SMTP, IXIA, TFTP, IXIA, Superflow, HTTP, TFTP
Worms	worms

Table 3: Feature distribution according to groups for NSL-KDD Cup data set

Group Name	**Feature No. of NSL-KDD data set**
Fundamental	1, 2, 3, 4, 5, 6, 7, 8, 9
Payload Dependent	10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22
Time Dependent	23, 24, 25, 26, 27, 28, 29, 30, 31
System	32, 33, 34, 35, 36, 37, 38, 39, 40, 41
Data Dependent	42, 43

Table 4: Feature distribution according to groups for UNSW-NB15 data set

Group Name	**Feature No. of UNSW-NB15 data set**
Basic	6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18
Content	19, 20, 21, 22, 23, 24, 25, 26
Time	27, 28, 29, 30, 31, 32, 33, 34, 35
General Purpose	36, 37, 38, 39, 40
Connection	41, 42, 43, 44, 45, 46, 47

are measured using MANN. In group elimination step, the groups are sorted based on their accuracy and the group with the lowest accuracy is eliminated and is not be considered in next iteration. In the feature selection step, the features from the eliminated group are evaluated and selected using genetic search algorithm. In MANN, we set the number of hidden layers to 1 to avoid vanishing gradient problem, learning rate to 0.3 and the number of training iteration to 50. Table 5 illustrates the accuracy for every iteration along with the group that is eliminated and features that are selected along for NSL-KDD Cup data set. Similarly, Table 6 shows the accuracy for every iteration along with the group that is eliminated and features that are selected for the UNSW-NB15 data set.

Table 5: Feature selection in each iteration of algorithm for NSL-KDD Cup data set

Iteration#	**Group Eliminated**	**Features Selected from Eliminated Group**	**Accuracy (%)**
1	Payload	10, 11, 12, 13, 14, 16, 17, 21, 22	85.71
2	Fundamental	2, 3, 4, 5, 6, 7	87.20
3	Time	23, 24, 25, 26, 29, 30, 31	88
4	System	33, 34, 35, 37, 38, 39, 40	90.36
5	Time	23, 24, 25, 26, 29, 30, 31	91.98

Table 6: Feature selection in each iteration of algorithm for UNSW-NB15 data set

Iteration#	**Group Eliminated**	**Features Selected from Eliminated Group**	**Accuracy (%)**
1	General Purpose	36, 37, 37, 40	89
2	Content	20, 21, 22, 23, 24, 25	90.41
3	Connection	42, 43, 44, 45, 46, 47	92.79
4	Time	27, 28 , 33, 34, 35	94.36
5	Basic	7, 8, 10, 11, 12, 13, 15, 16, 17	95.46

5.5 Training Phase

We used Microsoft Azure to build, and deploy the MANN in the completely observable cyber environment. Microsoft Azure will facilitate our training of supervised machine learning algorithm and attack detection in completely observable cyber systems. The machine learning algorithm was trained separately using two large data sets of network traffic, NSL-KDD, and UNSW-NB15 and then the trained MANN is used as web services in Microsoft Azure. This web service is used by the many applications connected to the cyber environment to detect attacks. In order to train the fully-connected MANN, we used back-propagation algorithm with initial learning weight diameter of 0.5 and a learning rate of 0.01. The binning normalizer is used as feature normalizer for a better result as it creates bins of equal size and values of each bin are normalized. The MANN is trained as fully connected with a number of hidden neurons as 46 and with learning iteration of 50. The full NSL-KDD and UNSW-NB15 training data set are used for the training the MANN. Finally, each trained algorithm is evaluated to find the accuracy of the detection of the attacks in the cyber systems.

5.6 Result

The full NSL-KDD and UNSW-NB15 test data sets are used to test and evaluate the trained MANN using the feed-forward algorithm.Table 7 and Table 8 demonstrate the number of attacks detected for each attack category respectively for NSL-KDD Cup and UNSW-NB15 data sets. We find that that attacks detected in the NSL-KDD and UNSW-NB15 data sets are promising. We found that the Exploits attack category in the UNSW-NB15 data set have low attack detection rate because content-based filtering technique was not used to extract the features of the UNSW-NB15 data set during the UNSW-NB15 data set generation [50].

Table 7: Number of attacks detected for each attack category of NSL-KDD Cup data set

Attack category	**Original number of attacks**	**Number of attacks detected**
DoS	7456	6738
Probe	2420	2210
R2L	2754	2210
U2R	200	50

Table 8: Number of attacks detected for each attack category of UNSW-NB15 data set

Attack category	**Original number of attacks**	**Number of attacks detected**
Analysis	2000	1052
Backdoor	1747	1037
DoS	12665	9341
Exploits	33393	19917
Fuzzers	18185	13358
Generic	40001	39212
Reconnaissance	10492	7301
Shellcode	1134	748
Worms	130	94

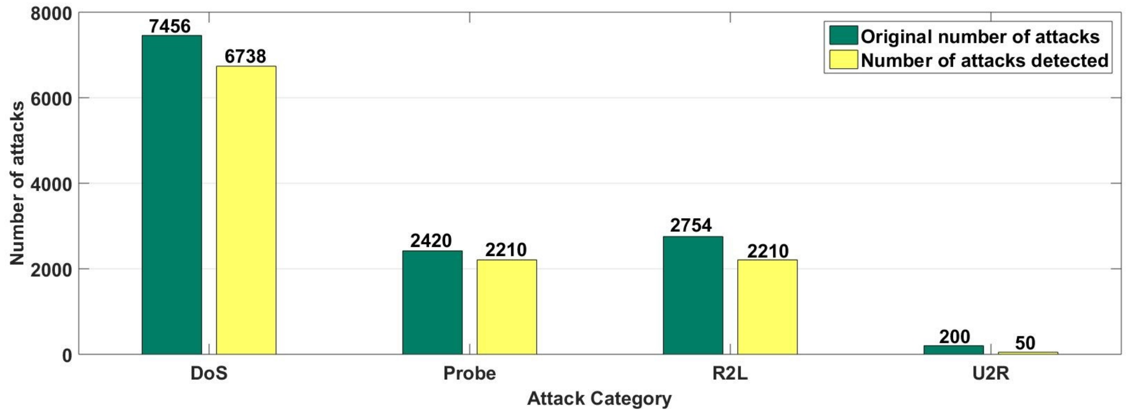


Figure 5: Number of attacks detected for each attack category of NSL-KDD Cup data set

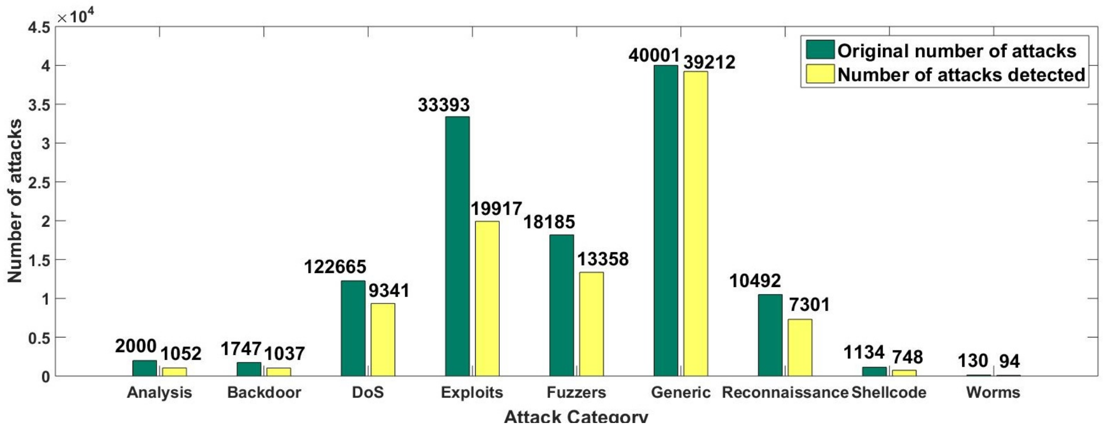
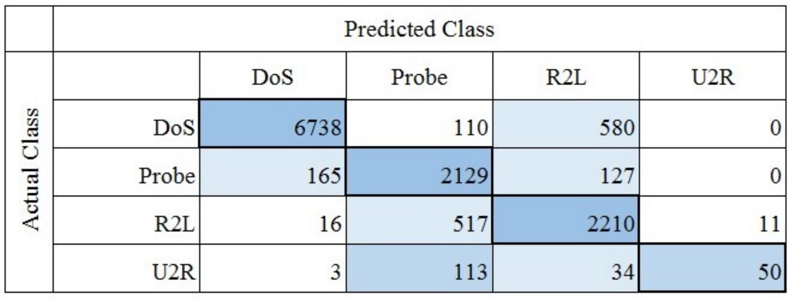


Figure 6: Number of attacks detected for each attack category of UNSW-NB15 Cup data set

5.7 Evaluation and Comparison

In the evaluation, the results of classification technique are compared with standard metrics are evaluated. It helps to evaluate the efficiency of trained MANN. Figure 7 shows the confusion matrix for NSL-KDD Cup dataset which describes the performance the trained classification algorithm.



		Predicted Class			
		DoS	Probe	R2L	U2R
Actual Class	DoS	6738	110	580	0
	Probe	165	2129	127	0
	R2L	16	517	2210	11
	U2R	3	113	34	50

Figure 7: Confusion Matrix for NSL-KDD Cup data set

The performance of our attack detection technique is compared to that of six most recent techniques. These six techniques are designated as follows: based multimodal artificial neural network (MANN) [22], multimodal logistic regression (LR) [24], multimodal decision forest (DF) [24], multimodal decision tree (DT) [17], multimodal Naïve Bayes (NB) [17] and multimodal support vector machine (SVM) [15]. Our results have been compared to the results of these techniques using the same data sets. As shown in Tables 9 and 10, our approach has substantial better performance than these existing techniques.

Table 9: Comparison of attack detection technique for NSL-KDD Cup data set

Existing Approaches on technique for NSL-KDD Cup data set			Our Approach
Techniques	Existing Approaches	Accuracy(%)	Accuracy(%)
MANN	Singh & Navjit [21]	81.20	91.98
LR	-	-	91.54
DF	-	-	91.11
DT	Tavallaee & Bagheri [17]	81.05	89.65
NB	Tavallaee & Bagheri [17]	76.56	85.70
SVM	Huy A. N. [15]	81.38	90.94

Table 10: Comparison of attack detection technique for UNSW-NB15 data set

Existing Approaches on technique on UNSW-NB15 data set			Our Approach
Techniques	Existing Approaches	Accuracy(%)	Accuracy(%)
MANN	Nour & Jill [22]	81.34	95.49
LR	Nour & Jill [22]	83.15	91.70
DF	-	-	90.76
DT	Nour & Jill [22]	85.56	92.90
NB	Nour & Jill [22]	82.07	90.12
SVM	-	-	94.44

Chapter 6

OVERALL APPROACH TO ESTIMATING TYPES OF STATES OF CYBER SYSTEMS IN PARTIALLY OBSERVABLE ENVIRONMENTS

In this chapter, our overall approach for estimating the types of states of a cyber system in the partial observable environments is presented. In this thesis, a type of state of the cyber system is fully identified by values of a suitable set of features of network traffic [30] of the cyber system. Let $F = \{f_1, f_2, ..., f_n\}$ be the set of features of the network traffic and $f_i = (f_{i1}, f_{i2}, f_{i3}, ..., f_{in})$ be the i^{th} instance of F, the state of the cyber system is defined by $s_k = (f_{a1}, f_{a2}, f_{a3}, ..., f_{aj})$. Each cluster c_j of the instances (state of the cyber system) $c_j = \{s_k, k = 1, 2, 3....\}$ generated by a selected clustering algorithm (such as K-means clustering algorithm) is defined as a type of states of the cyber systems. The network traffic raw data (packet capture) [51] is used for estimating the types of states of the cyber systems as the network traffic data provides various information about the cyber systems. The network traffic data is pre-processed [52] to select the packet with a valid header and then various attributes of each packet are extracted. In feature extraction step, three types of features namely network-behavior, signal-based and entropy-based features are extracted by aggregating the packet-level data [53]. The unsupervised feature selection algorithm [54] is used to select the relevant features subset from the original features that contribute most to the formation of clusters. The number of types of states of the cyber systems is determined by the elbow method [55] as this method measures how closely each data point is matched to data within its cluster [56]. K-means clustering algorithm [57] is used to partition each data point into different clusters and it produces labeled data set of the network traffic based on the number of types of states of the cyber system. The labeled network traffic data set is split into training and testing data sets. Some data of one or multiple features of the testing data set are randomly deleted. The convolutional neural network (CNN) [58] is trained with complete training data set as CNN learns the features deeply and determine the amount of overlap of one feature over another [59]. The trained CNN is used to estimate the probability of the types of states of the cyber systems using incomplete testing data set. In Figure 2, the system diagram of the overall approach of estimating the types of state of the cyber system in the partially observable environments is presented.

In this section, the estimation of the types of states of the cyber systems in the partially observable environments is presented using machine learning techniques. In this thesis, a benchmark and comprehensive network traffic data set [23] is used to determine the number of types of states of the cyber systems and to estimate the types of states of cyber systems in the partially observable environments. The estimation of types of states of the cyber systems in the partially observable environments is done mainly in 2 steps as shown in Figure 2. Each step is sub-divided into various sub-steps.

- Step A) Determining the number of types of states of the cyber systems

  Step A.1) The network traffic data [23] is collected from the cyber systems in packet capture file format [51] as the network traffic records the various activities of the cyber systems. The network traffic can be captured using a packet sniffer tools [34] like TCPDUMP [34] and Wireshark [38]. The captured network traffic data is pre-processed to keep only the packets with a valid header. The corrupted network packets are discarded. Various attributes of each network packet like payload length, flags and time stamp [51] are extracted from the network traffic data.

Step A.2) Three types of features are extracted from the attributes of the packets of network traffic. Apart from network behavior related features, signal-based (DFT) and entropy-based features of the cyber systems are also extracted [53]. The features are selected using unsupervised feature selection algorithm to determine the features that contribute most to the formation of clusters. Multi-Cluster Feature Selection (MCFS) algorithm is used as unsupervised feature selection algorithm as it finds the relevant feature subset of the original features that facilitates clustering [60].

Step A.3) The number of clusters formed by the network traffic data set determines the number of types of states of the cyber systems. The number of types of states of the cyber systems is determined by elbow method [55]. K-means clustering algorithm [57] is used to label the network traffic data set based on the number of types of states of the cyber systems. K-means clustering algorithm makes sure that there is at least one instance in each cluster and clusters do not overlap with each other.

- Step B) Estimation of types of states of cyber systems in partially observable environments

  Step B.1) The labeled network traffic data set obtained in Step A) is split into training and testing data sets using cross-validation method [36]. Some data of one or multiple features are randomly deleted from the testing data set because in partially observable environments the features which are missing are not known.

  Step B.2) The estimation of types of states of the cyber systems is done using supervised machine learning algorithm. Convolutional neural network (CNN) [58] is used as a supervised machine learning algorithm for the estimation of the types of states of the cyber systems as CNN learns the features deeply and is capable of incremental learning.

Step B.3) The incomplete testing data set is fed into the trained CNN to estimate the probabilities of types of states of the cyber systems in the different types of partially observable environments.

In this thesis we focus on the following:

- Step A, Determining the number of types of states of the cyber systems.
- Step B, Estimating the types of states of the cyber systems in the partially observable environments.

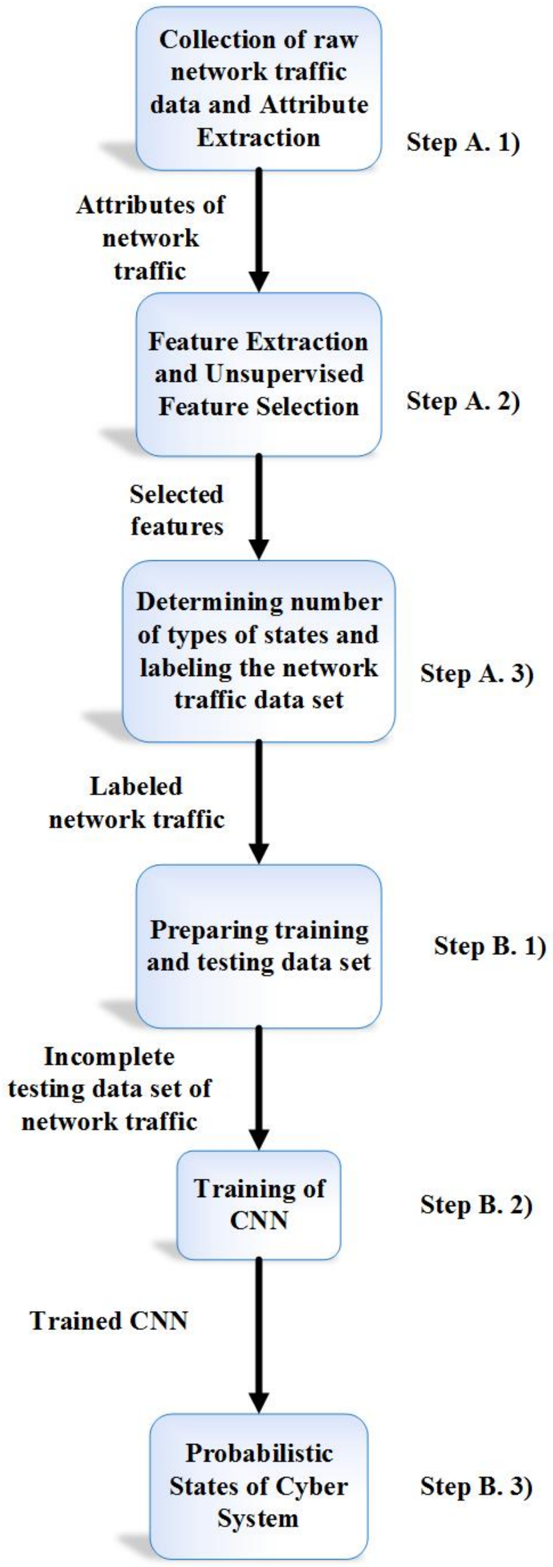

Collection of raw network traffic data and Attribute Extraction
Step A. 1)
Attributes of network traffic
Feature Extraction and Unsupervised Feature Selection
Step A. 2)
Selected features
Determining number of types of states and labeling the network traffic data set
Step A. 3)
Labeled network traffic
Preparing training and testing data set
Step B. 1)
Incomplete testing data set of network traffic
Training of CNN
Step B. 2)
Trained CNN
Probabilistic States of Cyber System
Step B. 3)

Figure 8: System diagram for estimating types of states of the cyber systems in partially observable environments

Chapter 7

ESTIMATION OF TYPES OF STATES OF CYBER SYSTEMS IN PARTIALLY OBSERVABLE ENVIRONMENTS

7.1 Overview

This chapter explains an approach used to estimate the types of states of the cyber systems in partially observable environments. In 7.2, we present the technique of capturing raw network traffic data and the technique to extract attributes from the packets of the network traffic data. In 7.3, we present an approach of extracting and selecting the features from the network traffic data. In 7.4, we present a technique to determine the number of types of states of the cyber systems and the technique to label the network traffic data set using K-means clustering algorithm. In 7.5, we present the technique of preparing the training and testing network traffic data set. In 7.6, we present the technique to train the convolutional neural network (CNN). In 7.7 we present the approach of estimating the probabilistic value of the types of states of the cyber systems in the partially observable environments.

7.2 Collection of Network Traffic Data and Attribute Extraction

Network traffic contains the data moving across the network of the cyber systems [61]. In the cyber systems, the network data is captured in the form of network packets and the network traffic data provide various information regarding the types of states of the cyber systems. The types of states of the cyber systems are useful in the traffic management [27], cyber state change identification [18], detecting attacks and predicting security breaches [3]. For example, if there is a lot of unusual traffic on a particular port and there is an unusual change in the types of cyber systems, it could possibly indicate the cyber attack. Thus, the change of types of cyber states due to the unusual activity of the cyber systems provide vital information about the cyber systems. Various packet sniffer tools like TCPDUMP [34] and Wireshark [38] are used to capture network traffic and these tools are also used for intercepting and displaying network packets on the network interface. The output of the captured network traffic is the packet capture (pcap) file which is handy in estimating the types of states of the cyber systems. The packet capture files contain various information about the cyber systems and parameters like source and destination IP address, protocols, and payloads [51] etc. These parameters provide information about the various types of states of the cyber systems. In this thesis, we shall use a large and comprehensive network traffic data [62] for estimating the types of states of the cyber systems.

The network traffic collected from the cyber system may be dirty in nature because of noise in the network traffic [63]. The raw network traffic data is cleaned and sanitized as dirty data can lead to poor state estimation of the cyber system. The network traffic data may also contain missing values, duplicates, typos [64]. Hence, the pre-processing of network traffic data is essential or else it produces poor results. During pre-processing, the raw network traffic data are converted into processed data of network traffic. This is done by selecting and filtering [65] of the parameters of each packet of network traffic. Each packet of the network traffic is processed and packets with valid header are selected. The corrupted packets of the network traffic with null payload lengths are also discarded. These corrupted packets are discarded and is not considered in the attribute extraction process.

The network traffic data has various quantitative and qualitative attributes about types of states of the cyber systems. The packet capture file of network traffic is in binary format and hence the attributes need to be extracted in human readable format. As network traffic data of cyber systems is encapsulated in the network packets, the attributes of each packet of network traffic are extracted. The attributes like source IP, destination IP, protocol, port, payload, and time-stamp [51] are extracted from each packet of the network traffic. In the pre-processing, the packet capture (pcap) file is provided as input to obtain various attributes of network packets. The attributes extracted from each packet of the network traffic data will be used to extract various types of features of the cyber systems.

7.3 Feature Extraction and Feature Selection

In this section, we present the technique of extracting features [66] of the cyber systems from network traffic data. As the estimation of types of states of the cyber systems is done for the partially observable environment, effective features are selected. In the feature extraction step, the conversations of the cyber systems are created by aggregating the packet-level data. Each conversation is identified by the source and destination IP address. Narang et al [53], presented that three types of features of network traffic are effective in detecting attacks and identifying changes in the cyber systems. In this thesis, three types of features of the cyber systems are extracted from network traffic data:

1. Network behavior based features: The network behavior based features provide details about the packet signatures of the network traffic. It also provides vital details about the different configurations of the cyber systems. The network

behavior based features help to estimate different types of states of the cyber system. The network behavior based features provide details related to cyber security, cyber traffic change, attack detection, security breach detection and prediction. The features like network traffic flow, total payload, payload sent, payload received and inter-arrival time etc. [53] are extracted as network behavior based features as these features provide information about the cyber systems.

2. Signal based features: Discrete Fourier Transformation (DFT) [67] is applied on the payload and the inter-arrival time of each packet in a conversation of the network traffic [53]. DFT captures hidden information patterns of the cyber systems; thus, the signal based featurse are useful in estimating the types of states of the cyber systems in the partially observable environments [61]. Given a time sequence $X = X(0), X(1), X(2)......, X(w-1)$, its DFT is given as

$$DFT(X) = \frac{1}{\sqrt{w}} \sum_{k=0}^{w-1} X(k).e^{\frac{-2j\Pi kn}{w}}$$

In [68], it is presented that DFT on network traffic is effective in detecting anomalous traffic data from large-scale time series data that exhibit patterns over time. Most of the hidden information (energy) about the types of states of cyber systems is concentrated in the first few coefficients of DFT [69]. Moreover, DFT has an attractive property that the amplitude of Fourier coefficients is invariants under shifts. Thus, using DFT in feature extraction has potential that it can be extended to finding a similar sequence in features of the cyber systems. Thus, in this thesis, first three coefficients are evaluated to capture the hidden communication patterns of the cyber systems.

3. Entropy-based feature: The size of the payload of each packet of network traffic exhibits more variation when there is any unusual change in the states of the

cyber system [53]. In [70], the entropy-based feature is used to track changes in the cyber system. The variation is observed in payload size of the packet of the network traffic when there is an unusual activity or change in types of state of the cyber systems [53]. Hence, the entropy-based feature is implemented on the payload size of the network packet in order to calculate the amount of randomness or entropy in payload size variation. The unusual change in the types of states of the cyber systems will have high entropy (or lower compression size) than the normal activity of the cyber systems. The compressed size of the payload is determined based on the optimal encoding limit established by Shannon. The expected payload length L of an encoding of X with associated probability p(x) is given by [71]:

$$L(x) = \sum_{x \in X} p(x) \log_2(p(x))$$

Feature selection is one of the important steps in estimating types of states of the cyber systems as feature selection helps in reducing high-dimensional data for machine learning problem. Feature selection also helps to reduce the computational complexity of the machine learning algorithms [72]. Due to lack of label information of the network traffic data set, the evaluation of the contribution of features for estimating types of states of the cyber systems is done using unsupervised feature selection algorithm. The unsupervised feature selection algorithm selects the subset of features from the network traffic data set. The unsupervised feature selection algorithm is designed for clustering problem and seek an alternative criterion to define the relevance of features such as data dissimilarity and local discriminative information [73]. In this thesis, Multi-Cluster Feature Selection (MCFS) algorithm [60] is selected as unsupervised feature selection algorithm because MCFS guides in selecting features without the class labels. MCFS facilitates to select the relevant features that contribute most to the formation of clusters, and MCFS consider the possible correlation between different features and produce optimal feature subset [72]. Thus, MCFS is effective in selecting features from the network traffic data because it will consider the correlation between the features of the network traffic data. Moreover, MCFS select only those features from the data set that preserve the multi-cluster structure of the data and MCFS is more effective than the other existing unsupervised feature selection algorithms [60].

7.4 Determining the Number of Types of States and Labeling the Network Traffic Data Set

Determining the number of types of states of the cyber systems is one of the important steps in estimating the probabilistic state of the cyber systems. As the network traffic data set is not labeled, the number of types of states of the cyber systems are not known. The classifier algorithm is used to determine the number of types of states of the cyber systems. The classifier algorithm produces different clusters based on the data points of the network traffic data set. In the thesis, each cluster produced by the classifier algorithm is considered as types of state of the cyber system. Thus, each type of state of the cyber system is represented by each cluster of the clustering algorithm because each cluster represents the set of variables used to describe a type of state of the cyber system. In this thesis, elbow method [55] is used to determine the number of types of states of the network traffic data set. The computational complexity of elbow method is less compared to other existing methods and the elbow method has high accuracy and is based on the idea that adding another cluster to the data set does not yield better results [74]. This is important in practical applications in estimating the types of states of the cyber systems. The elbow method measures at the percentage of variance as a function of the number of clusters [75]. The method is based on the idea that one should choose a number of the clusters so that adding another cluster does not give better modeling of the data. The elbow method tries to minimize the total intra-cluster variation which is helpful in measuring the compactness of the clusters.

Initially, the elbow method computes the sum of squared error (SSE) for some value of k, where k represents the number of types states of the cyber systems. The SSE is the sum of the squared distance between each member of the cluster and its centroid. Thus, mathematically it can be represented by

$$SSE = \sum_{i=1}^{K} \sum_{x \in c_i} dist(x, c_i)^2$$

Thus, elbow method measures how closely the members is matched to data within its cluster and how loosely it is matched to data of the neighboring cluster. The value of SSE decreases with the increase of a number of types of states of the cyber systems because the distortion becomes smaller with the increase in the number of types of states of the cyber system. According to elbow method, the number of types of states of cyber systems is selected for the value of k where SSE decreases abruptly. The algorithm for elbow method is shown in the Algorithm 3.

Algorithm 3 Elbow Method to determine the number of types of states of the cyber systems

```
Require: Initialize $k$
1: repeat
2:     Increment the value of k
       Measure the value of SSE
3: until The value of k where the value to SSE drops dramatically
4: Select that $k$
```

A classifier algorithm is used to label each observation of the network traffic data set into a number of types of states. In the thesis, K-means clustering algorithm [57] is used as a classifier algorithm which divides the observations of the network traffic data set into k number of clusters. Each cluster is considered as a state of the cyber system. K-means clustering algorithm aims to minimize the intra-cluster sum of squares (SSE) and partition data points into k clusters so that each observation belongs to cluster with the nearest mean. Thus, K-means clustering algorithm helps to form types of states of the cyber systems based on each observation of the network traffic data set. Each data point of network traffic data set is labeled according to the clusters formed by the K-means clustering algorithm. In K-means clustering algorithm, k stands for the number of clusters and it is a user input to the algorithm. We have determined the value of k in the previous step using the elbow method. K-means clustering algorithm is chosen over the other existing data clustering algorithms (like hierarchical clustering algorithm and expectation maximization clustering algorithm) as k-means clustering algorithm is ideal for huge data set and large number of clusters and it is computational complexity is less compared to other existing clustering algorithms [76].

K-means clustering algorithm is one of the simplest unsupervised machine algorithm used in clustering problems. In K-means clustering algorithm, the k number of centroids are defined, one for each cluster and the centroids are placed far from each other. The K-means algorithm associates each observation of the network traffic data set to the nearest centroid. K-means clustering algorithm then re-calculate k new centroids and binding is done between the same observations of the network traffic data set and the nearest new centroid. This continues till there is no more change in the position of the centroid. Thus, K-means clustering algorithm aims to minimize the value of the squared error function, given by

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} |x_i^j - c_j|^2$$

, where $|x_i^j - c_j|^2$ is the distance between the observation $|x_i^j|$ and cluster center $|c_j|$. Algorithm 4 describes the steps of the K-means clustering algorithm.

Algorithm 4 K-means clustering algorithm

```
Require: Initialize $c_1, c_2, ....., c_k \in \mathbb{R}^n$
1: repeat
2:     For every $i$, set $c^{(i)} = argmin_j ||x_i^j - c_j||^2$
       For each $j$, set $c_j = \frac{\sum_{i=1}^{m}(c^{(i)}=j)x^{(i)}}{\sum_{i=1}^{m}(c^{(i)}=j)}$
3: until until convergence
4: Select that $k$
```

K-means clustering algorithm always makes sure that there are k number of types of states during the estimation of types of states for the cyber system and there is at least one observation in each types of state. The types of states of the cyber system, which is determined by the K-means clustering algorithm make sure that the type of states do not overlap with each other. Hence, there is no ambiguity with the existence of same observation in two types of states of the cyber system, thus making sure that each observation shares only one type of state of the cyber system.

7.5 Preparing Training and Testing Data Set of Network Traffic

The labeled data set is used as the training and testing data set for training and testing machine learning algorithm respectively. Splitting the labeled network traffic data set into training and testing data sets is an important step in the estimation of the types of states of the cyber systems. Typically, when splitting the data set into the training and testing data set, the training data set is larger in size compared to the testing data set. The data set is split into training and testing data set using K-fold cross validation method [36] where the data set is randomized into K equal size partitions. The K^{th} partition is selected as a testing data set and $K - 1$ partitions are selected as the training data set. K-fold cross validation is used to split the data set into training and testing data set as it helps in unbiased classification results and there is no overlapping between training and testing data set [36].

In this thesis, the complete training network traffic data set is used to train the machine learning algorithm. If the training data set is incomplete, it is made complete by replacing the missing values by either one of the method:

1. Replace with mean
2. Replace with median
3. Replace with mode

The completeness of the incomplete training data set will increase the effectiveness of estimating the types of states of the cyber systems in partially observable environments. But we shall use incomplete testing data set to estimate the types of states in cyber systems in partially observable environments.The machine learning algorithm is used as a classifier to estimate the probabilistic values of the types of states of the cyber system. As the estimation of the types of states of the cyber system is performed in the partially observable environments, the testing data set is made incomplete. The incompleteness of the testing data set is performed by randomly selecting each observation of testing data set and deleting the value. The making of the incomplete testing data set is performed in following ways:

1. Randomly selecting one feature of each observation of the network traffic data set.
2. Randomly selecting two features of each observation of the network traffic data set.
3. Randomly selecting three features of each observation of the network traffic data set.
4. Randomly selecting few data points of one feature of the network traffic data set.
5. Randomly selecting few data points of two features of the network traffic data set.
6. Randomly selecting few data points of three features of the network traffic data set.

The complete training data set will be used to train the machine learning algorithm and the incomplete testing data set to estimate the probabilistic value of the types of states of the cyber systems in the partially observable environments.

7.6 Training of Convolutional Neural Network

Deep Learning has attained significant outcomes in the field of computer vision and speech recognition in recent years. In this thesis, convolutional neural network (CNN) is used as deep learning algorithm to estimate the types of states of cyber systems in the partially observable environments. CNN is the feed forward deep neural network which includes some convolutions in some layers [77]. A convolutional neural network is composed of one or more convolution layers, a sub-sampling layer followed by one fully connected layer. In convolution layer, the convolution operation is applied where the parameters are reduced by weight sharing technique [77]. In sub-sampling layer, non-linear function and pooling operation are applied for distortion-invariant features. Thus, CNN leverage the idea of local connectivity, parameter sharing and pooling [78]. The convolution layer produces feature maps by linear convolution filter followed by a nonlinear activation function [77]. The feature map can be given by:

$$f_{i,j,k} = max(w_k^T x_{i,j}, 0)$$

where (i, j) is the index of the feature map, x_{ij} stands for the input patch centered at the location (i, j) and k is used to index the channel of the feature map. For classification task, softmax is applied as loss function in the output layer and the output layer has one neuron per class.

Traditionally, 2-D CNN is used [79] for image classification; but in this thesis 1-D CNN is used for classification to estimate the types of states of the cyber systems in the partially observable environments. Back-propagation algorithm is used for efficient training of the 1-D CNN. In 1-D CNN, 1-D arrays are used in each neuron for its kernels, input and output variables for both forward propagation [80] and back-propagation algorithm [13]. Thus, 1-D array operations like 1-D convolution and reverse are performed instead of 2-D operations. Also, the kernel size and sub-sampling [81] are used as single scalar values. The forward propagation of 1-D CNN is presented by the below equation

$$x_k^l = b_k^l + \sum_{i=1}^{N_{l-1}} con1D(w_{ik}^{l-1}, s_i^{l-1})$$

where $con1D()$ represents the 1- convolution, x_k^l is the input, b_k^l is the bias of the $k^t h$ neuron at layer l, s_i^{l-1} represent the output of the i^{th} neuron for layer $l-1$ and w_{ik}^{l-1} denotes the kernel from i^{th} neuron of layer $l-1$ to the k^{th} neuron of layer l [81].

In 1-D CNN, the delta error Δs_k^l for back-propagation is represented by the below equation:

$$\Delta s_k^l = \sum_{i=1}^{N_{l+1}} con1Dz(\Delta_l^{l+1}, rev(w_{ki}^l))$$

, where $rev(.)$ reverse the array and $conv1Dz(.,.)$ executes full convolution in 1-D [13].

In CNN, the individual neurons are arranged in such a fashion that they respond to the overlapping regions of features. This helps in the estimating types of states of the cyber systems when the environments are partially observable. CNN learns deeply the features of the cyber systems and expresses the amount of overlap of one feature of network traffic over another; thus measuring the similarity of two features of network traffic [59]. CNN can be trained easily and is computationally efficient as they do not need specialized hardware for implementation [82]. Moreover, CNN has the capability of online incremental learning; this is more effective in estimating types of states for cyber systems where the environments are partially observable.

In this thesis, the number of nodes in the input layer of CNN is same as the number of features selected by MCFS algorithm. The number of nodes in the output layer of CNN is the same as the number of types of states in the cyber system. Our CNN has two hidden layers, out of which the one which is connecting to the input layer is convoluted; and another is fully connected to the output layer. The number of nodes in two hidden layers are same as the number of nodes in the input layer. The learning rate for the CNN is set to 0.01 since higher learning rate can over shoot the local minima. The initial weights of the edges of the input layer of the CNN is assigned to 0.9 as larger weights will slow down the process of learning of nodes of CNN. The learning iteration of the ANN is set 20 because higher learning rate will slow down the training process. In the output layer, softmax function is used as softmax function is the gradient log normalizer and it is used in various probabilistic multiclass classification methods. Moreover, softmax function can be used with both discrete and continuous data [83].

7.7 Estimation of Types of States of Cyber Systems in Partially-Observable Environments

The complete training network traffic data set is used to train the 1-D CNN as this will facilitate the estimation of the types of states of the cyber systems in the partially observable environments. There are six different kinds of testing data set of network traffic are used to estimate the types of states of the cyber systems. In this thesis, the analysis of the accuracy for classification of the trained 1-D CNN is done and the results of the classification are compared with other existing multimodal classifier algorithms. In the evaluation, the result of the classification technique is evaluated with the standard metrics like accuracy, recall and precision. This helps to measure the efficiency of the trained 1-D CNN. The probabilistic values of types of states of the cyber system are estimated based on the results of the classification done by CNN. The probabilistic values of types of states of the cyber system are expressed by

$$P(s_i) = \frac{\sum_{i=1}^{i} x_i}{\sum_{n=1}^{n} X_n}$$

, where $\Sigma_{i=1}^{i} x_i$ represents the total number of expected observation for state i as estimated by 1-D CNN and $\Sigma_{n=1}^{n} X_n$ is the total number of observations in the testing data set.

Chapter 8

CASE STUDY OF ESTIMATING TYPES OF STATES OF CYBER SYSTEMS IN PARTIALLY OBSERVABLE ENVIRONMENTS

8.1 Overview

In this chapter, a case study is conducted to estimate types of the states of the cyber systems in the partially observable environment. A large and comprehensive network traffic data set is used for the case study. In 8.2, the detailed description of the network traffic data used in this thesis and the technique to extract attributes from the packets of network traffic. The techniques for feature extraction and feature selection are presented in 8.3. In 8.4, the mechanism to determine the number of types of states of the cyber system and the technique to label network traffic data set using unsupervised clustering algorithm is presented. The technique for generating the training and testing data set in shown in section 8.5. In 8.6, the convolutional neural network is trained using the training network traffic data set. Finally, in 8.7 the results for estimating types of states of the cyber systems is presented and our approach is compared with other existing machine learning algorithms.

8.2 Overview of Network Traffic Data Set and Attributes Extraction

In the thesis, UNB ISCX Intrusion Detection Evaluation data [23] is used to estimate the types of states of the cyber systems in the partially observable environment. The network traffic data is generated using a systematic approach which includes dynamic and evolving network behaviors and patterns. The network traffic data was generated by Information Security Centre of Excellence (ISCX) of University of New Brunswick and consists of data for a period of 7 days of network activity (normal and malicious). The network traffic data is selected for estimating the types of states of the cyber systems because the network traffic data processes the following characteristics:

1. The network traffic data consists of the realistic network traffic and does not contain any artificial post-capture trace insertion.
2. The network traffic data contains total interaction capture which is essential in estimating states of the cyber system.
3. The network traffic provides complete capture including privacy settings related to sharing real network traces.
4. The network traffic data includes diversified attack scenarios and has considered new types of complex attacks.
5. The network traffic data includes real traffic for HTTP, SMTP, SSH, IMAP, POP3, and FTP.

The network traffic data was generated using 21 interconnected Windows workstations as a diverse set of known vulnerabilities can be generated in Windows operating systems [23]. The workstations are divided into 5 LANs consists of real internetwork connections, web server, email, Domain Name System (DNS) and Network Address Translation (NAT).

Network traffic captures the information moving across the network at a given point of time. The network traffic contains encapsulated network packets and it provides various information about the states of the cyber system. The network traffic can be captured using Wireshark [38] which is a network analysis tool. The tool helps to capture the network packets by sniffing on the interface(s) of various connected devices of the cyber system [34]. The output of the capture network traffic is the packet capture (pcap) files which are handy in estimating the probabilistic value of the types of states of the cyber system. The packet capture file contains information about the source IP, destination IP, protocols, and payloads etc [51].

In this module of attribute extraction, the captured network traffic i.e. the packet capture files are processed to remove the corrupted packets and the attributes are extracted from the network traffic data. The module, which obtains the packet-level data from the packet capture file was developed in Python 2.7 [84] and TShark [53] on Amazon Web Service [85] E2 instance (m4.4xlarge) running Ubuntu 16.0 OS. The program iterates over all the captured network traffic files and extract the different attributes of each network traffic and save it as a comma separated values (CSV) file. In order to pre-process the captured network traffic, the packets with the valid TCP/UDP header are kept. The program ignores those packets where if both TCP and UDP payload lengths are null and subtract 8 bytes from UDP payload to account for UDP header. The program then extracts the attributes of each packets using the TShark [86] which is later used to extract various types of features of the network traffic data. For each packet, Table 11 lists the details of the attributes [53] extracted from the network traffic data. The payload length and the flags depend on the protocol (TCP or UDP).

8.3 Feature Extraction and Feature Selection

In this module, the features are extracted based on the packet-level information as described in Section 8.2. Like before, the module was developed using Python 2.7 [84] on Amazon Web Service [85] E2 instance (m4.4xlarge) running Ubuntu 16.0 OS where

Table 11: List of attributes extracted from each packet

Attribute Name	**Description**
Source IP	The IP address of the source
Destination IP	The IP address of the destination
Protocol	The field denotes the current IP protocol begin used
Frame Time Epoch	Epoch time
TCP Length	The length of TCP segment data
UDP Length	The length of UDP segment data
TCP Flags	The flags of TCP packet
TCP Flags Reset	Control bit to reset the connection if the sender encountered any problem
TCP Flags Syn	Control bit to synchronize sequence number and initialize the connection
TCP Flags Fin	The sender of the segment request to close the connection
TCP Flags Ack	The acknowledgment bit is set to 1 when this segment is carrying on an acknowledgment.
TCP Flags Urg	This control bit is set to 1 when the priority data transfer feature has been called.
TCP Flags Push	This denotes that the data in the segment is pushed immediately on the application of the receiving end.
TCP Source Port	Port number for the TCP source port
TCP Destination Port	Port number for the TCP destination port
UDP Source Port	Port number for the UDP source port
UDP Destination Port	Port number for the UDP destination port

the output of the pre-processing and attribute extraction module is fed as input. The features are extracted from each flow which is identified by source and destination IP address and the maximum inter-arrival time taken between two packets. This module extracts three types of features – network behavior-based features, signal-based features, and entropy-based feature. Discrete Fourier Transformation [87] was applied on the payload length of each packet and inter-arrival time for each flow to extract the signal-based features. The entropy-based feature was extracted by applying optimal encoding limit on the payload size of each packet. Table 12 demonstrate the list of 35 features [53] that are extracted along with their descriptions.

After the features are extracted from the UNB ISCX Intrusion Detection Evaluation data, statistical analysis is performed on the result of the output in R programming [88] at AWS EC2 instance. Thus, the data set has 31 features and there are 1,951,464 connection links of network traffic. The summary of the network traffic data set is presented in Table 13.

Principal Component Analysis (PCA) [89] is performed on the network traffic data set to extract important information from the multivariate network traffic data set and to direct a set of new features called principal components. PCA helps to find directions where the variation of the network traffic data set is maximal. Figure 9 describes the importance of principal components in the form of scree plot. It shows that the first five principal components hold 61% (approx.) of the variances.

Table 12: List of extracted features

Feature Type	**Feature Name**	**Description**
network-based features	Flow duration	Duration of each flow
	Sender payload	Total payload size for sender
	Receiver payload	Total payload size for receiver
	First packet size	The size of the first packet
	Maximum packet size	The maximum size of packet in each flow
	Median inter-arrival time	Median of total inter-arrival time in each flow
	Inter-arrival sent	Total inter-arrival time for sending
	Inter-arrival received	Total inter-arrival time for receive
	Packets sent	Total number of packets sent
	Packets received	Total number of packets received
	Average payload	Average payload in each flow
	Variance payload	Variance of total payload
Entropy-based features	Compression	Compression ratio
Signal-based features	Prime wave magnitude payload	The magnitude of the prime wave for the payload
	Prime wave phase payload	The phase of the prime wave for the payload
	DFT payload magnitude 1-3	The first three magnitude coefficient of DFT for the payload
	DFT payload phase 1-3	The first three phase coefficient of DFT for the payload
	DFT variance magnitude payload	The variance of magnitude of the payload for the DFT
	Prime wave magnitude inter-arrival time	The magnitude of the prime wave of the inter-arrival time
	Prime wave phase inter-arrival time	The phase of the prime wave for the inter-arrival time
	DFT inter-arrival time magnitude 1-3	The first three magnitude coefficient of DFT for the inter-arrival time
	DFT inter-arrival time phase 1-3	The first three phase coefficient of DFT for the inter-arrival time
	DFT variance magnitude inter-arrival time	The variance of magnitude of the inter-arrival time for the DFT

Table 13: Summary of the network traffic data set

Feature Name	**Minimum value**	**Median value**	**Mean value**	**Maximum value**
Flow Duration	0.0000	0.0428	21.4272	1999.9877
Sender Payload	0.0000	72.0	909.6	75416.0
Receiver Payload	0.0000	51	2097	42340
First Packet Size	0.0000	1380	1051	1460
Maximum Packet Size	0.0000	1430	1122	1460
Median Total Inter Arrival Time	0.0000	0.0000	0.0242	873.9208
Inter Arrival Sent	0.0000	0.0000	0.0242	873.9208
Inter Arrival Receive	0.0000	0.0000	0.0006937	1.3202727
Packets Sent	1.0000	1.0000	1.359	68.000
Packets Received	3.0000	3.0000	3.512	40.000
Average Payload	0.0000	1380.0	1059.7	1460.00
Variance Total Payload	0.0000	0.0000	34312	532170
Compression Ratio	2.526	12.202	17.479	480.0
Prime Wave Magnitude Payload	0.0000	108	2703	94900
Prime Wave Phase Payload	-9e+01	0e+00	9e-03	9e+01
DFT Payload Magnitude 1	0.0000	1764	3007	75416
DFT Payload Phase 1	-84.000	0.0000	0.02089	90.0000
DFT Payload Magnitude 2	-90.000	0.0000	-0.4513	90.000
DFT Payload Magnitude 3	0.0000	1460	1324	123214
DFT Payload Phase 3	-90.000	0.000	1.57	90.000
Variance Magnitude DFT Payload	0.0000	1460	1433	3265
Prime Wave Magnitude Inter Arrival Time	0.0000	0.0000	0.0394	1747.8417
Prime Wave Phase Inter Arrival Time	-90.000	0.000	2.706	90.000
DFT Inter Arrival Time Magnitude 1	0.0000	0.0000	0.0582	1747.8417
DFT Inter Arrival Time Phase 1	-60.000	0.000	-14.42	0.00
DFT Inter Arrival Time Magnitude 2	0.0000	0.0000	0.01749	1747.6688
DFT Inter Arrival Time Phase 2	-90.00	0.000	2.706	90.000
DFT Inter Arrival Time Magnitude 3	0.0000	0.0000	0.0121	1747.6688
DFT Inter Arrival Time Phase 3	-90.00	0.00	11.71	90.00
Variance Magnitude DFT Inter Arrival Time	0.0000	0.0000	0.0574	1747.8417

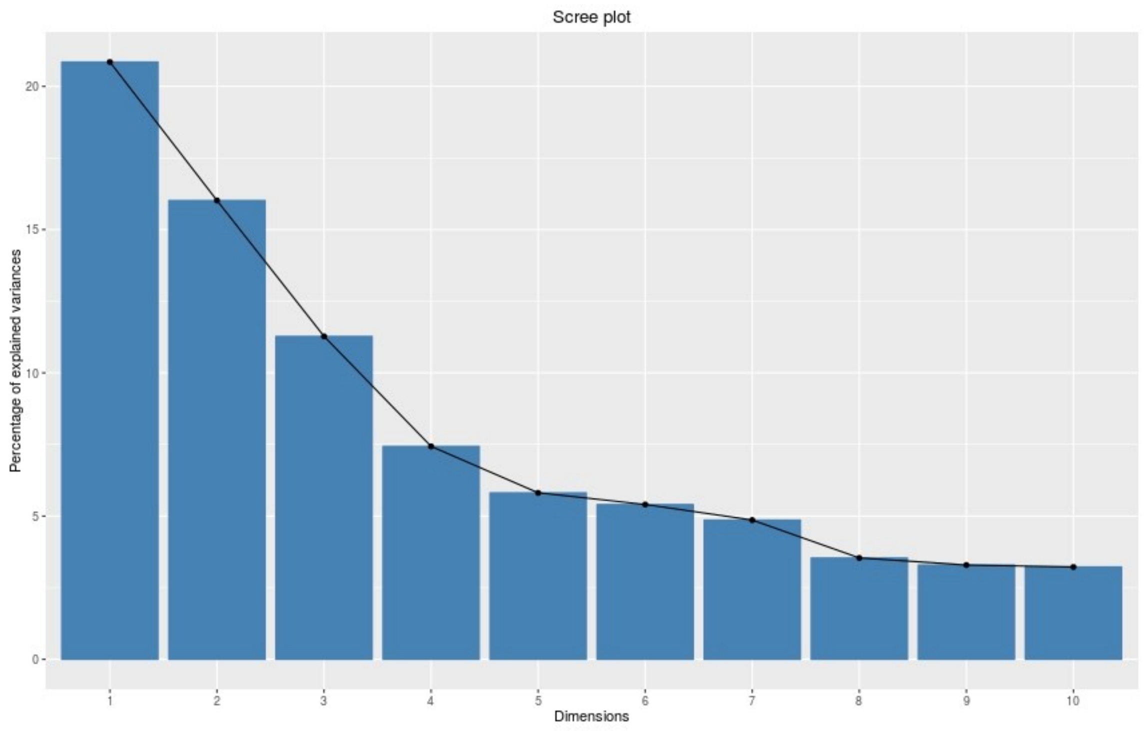


Figure 9: PCA Scree Plot of UNB ISCX Intrusion Detection Evaluation data set

The features that are correlated with the principal components 1 to 5 plays an important role in the variability of the data set. Figure 10 shows the features that are associated with the first five principal components and they contribute more to the components.

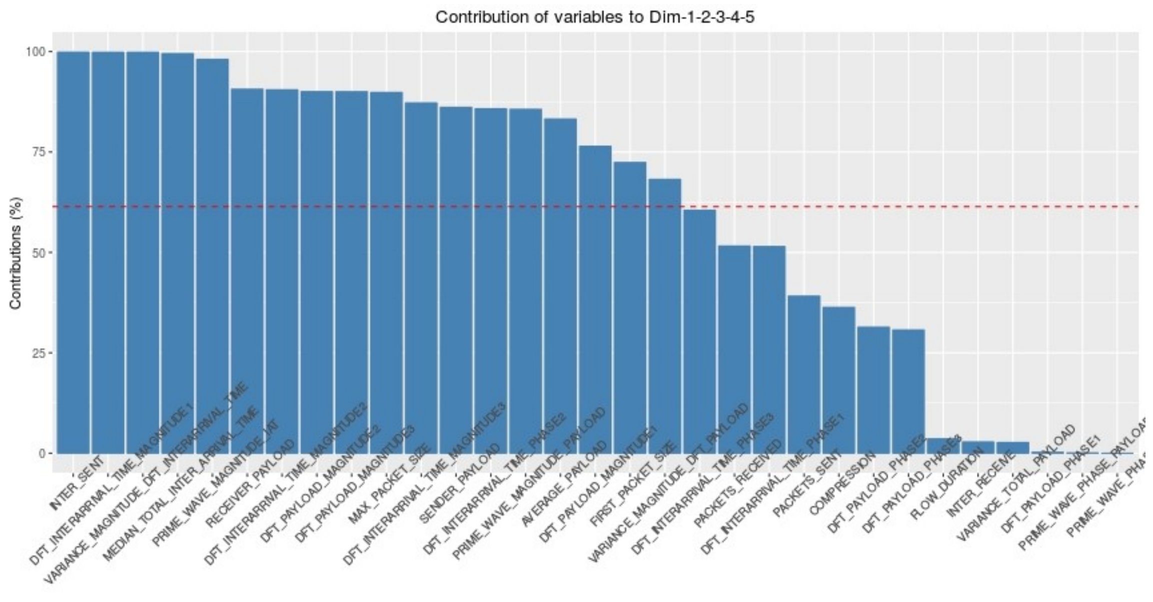


Figure 10: Contribution of features of UNB ISCX Intrusion Detection Evaluation data set to principal components

The unsupervised feature selection algorithm is applied to the network traffic data set in order to select the features that contribute most to the formation of the clusters. Multi-cluster feature selection (MCFS) [60] algorithm is selected as unsupervised feature selection algorithm as it selects the features that cover the multi-cluster structure of the data and it uses spectral analysis to correlate between the features. MCFS algorithm is implemented in Python 2.7 using the NumPy [90], SciPy [91], and Scikit-learn [84] packages in AWS EC2 instance (m4.4xlarge) running Ubuntu 16.0 OS. At first, the feature selection module constructs an affinity matrix and then feature weight matrix are obtained. Finally, the algorithm sorts the features based on the feature scores of feature weight matrix. Table 14 shows the list of features selected by the MCFS algorithm along their feature scores.

Table 14: Features selected by MCFS algorithm

Selected Features	**Feature Scores**
Sender Payload	1.64779830e+04
Receiver Payload	1.64779830e+04
Median Total Inter Arrival Time	3.23631675e-01
DFT Payload Magnitude (1st Co-efficient)	3.22043241e-01
DFT Inter Arrival Time Magnitude(1st Co-efficient)	9.43615546e+04
DFT Inter Arrival Time Phase(1st Co-efficient)	5.16977409e+05
DFT Inter Arrival Time Phase(2nd Co-efficient)	5.16977409e+05
DFT Inter Arrival Time Magnitude(3rd Co-efficient)	2.84661113e-01
DFT Inter Arrival Time Phase(3rd Co-efficient)	5.16977409e+05
Variance Magnitude DFT Inter Arrival Time	5.88219974e-01

8.4 Determining Number of Types of States and Labeling the Network Traffic Data

The number of clusters formed by the network traffic data set represents the number of types of states of the cyber system. In this thesis, partitioning method i.e. K-means clustering algorithm is used to label the network traffic data set based on the number of types of states of the cyber system. The optimal number of clusters need to be determined as a number of clusters is provided as input to K-means algorithm [57]. Elbow method [55] is used to determine the number of types of states of the cyber system as it applies a direct method which involves optimizing a criterion called cluster sum of squares (SSE). R programming is used to implement the elbow method for K-means clustering algorithm in AWS EC2 instance. Figure 11 shows the output of the elbow method and the graph suggests eight types of states for the cyber system. The graph obtained from the elbow method shows that at the value of k suddenly drops at the value of 8 and forms an “elbow” shape.

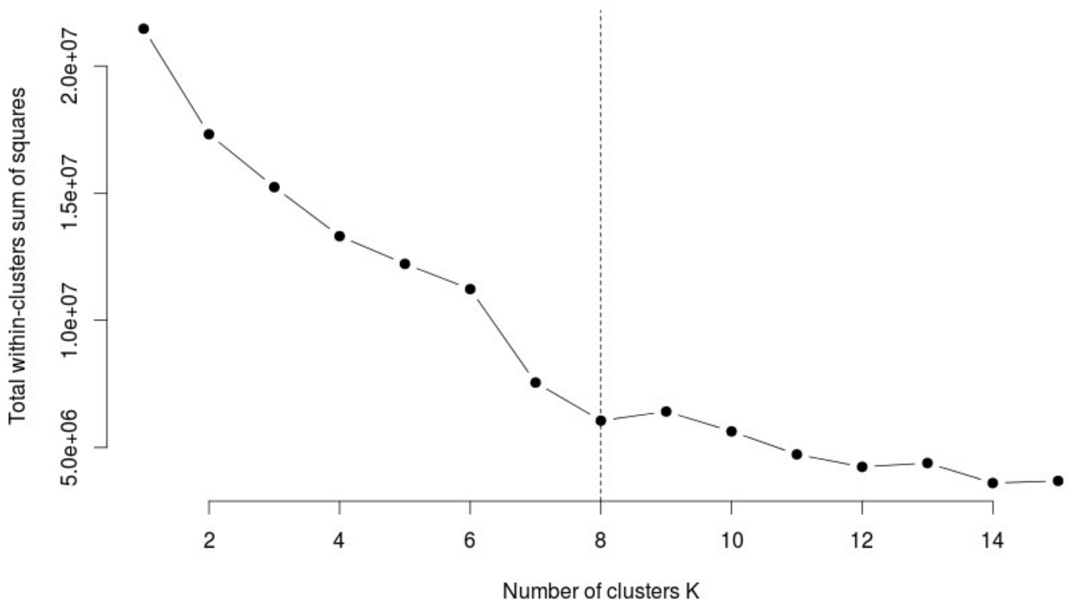


Figure 11: Number of types of states of cyber system using Elbow method

K-means clustering algorithm is used to label the network traffic data set as K-means is one of the simplest clustering algorithms to estimate the types of states of the cyber system. As the network traffic data set does not contain a label column, K-means algorithm is suitable as it uses unsupervised learning method. K-means algorithm uses iterative procedures to cluster similar characteristics in a data set. This helps to identify each observation of the network traffic data set into distinct types of states. The number of types of states of the cyber system is already determined by Elbow method. So the number of initial centroids is set to 8. In K-means clustering algorithm, the data points are randomly placed in a cluster and then the initial means of a Euclidean distance of the randomly assigned data point from the centroids are computed. The random number seed is set as 123456789 as it significantly affects the degree of randomness of the initialization. The K-means clustering algorithm is executed in Microsoft Azure Machine Learning Studio and Microsoft Azure for efficient and effective computation. Table 15 describes the detailed configurations used in K-Means clustering algorithm. The output of this module labels each data points of the network traffic data set according to the number of types of states of the cyber system.

Table 15: Configuration used in K-Means clustering algorithm

Description	**Configuration Details**
Number of Centroids	8
Initialization	Random
Random Number Seed	123456789
Metric	Euclidean
Number of Iterations	1000

8.5 Preparing Training and Testing Data Set

The labeled network traffic data set is split into training and testing data set using k-fold cross-validation in order to overcome the problem of over-fitting [59]. The 10-fold cross-validation is performed on the network traffic data set to split into training and testing data set. The splitting of the network traffic data set using 10-fold fold cross validation is executed in R programming in AWS EC2 instance running Ubuntu 16.0 operating system.

If the training data set is incomplete, it is made complete by replacing the missing values by either one of the method:

1. Replace with mean
2. Replace with median
3. Replace with mode

The completeness of the incomplete training data set will make increase the effectiveness of estimating types of states of the cyber systems in partially observable environments. But we shall use incomplete testing data set to estimating the types of states in cyber systems in partially observable environments.

In the thesis, the types of states of the cyber system are estimated in the partially observable environments. Hence, the testing data using to estimate the probabilistic values of types states of the cyber system is incomplete. A JAVA program is used to convert the testing data incomplete. There are six different types of incomplete testing data sets are used to estimate the types of states of cyber system in six types of partially observable environments:

- Type 1: Randomly selecting one feature of each observation of the network traffic data set
- Type 2: Randomly selecting two features of each observation of the network traffic data set
- Type 3: Randomly selecting three features of each observation of the network traffic data set
- Type 4: Randomly selecting few data points of one feature of the network traffic data set
- Type 5: Randomly selecting few data points of two features of the network traffic data set
- Type 6: Randomly selecting few data points of three features of the network traffic data set

8.6 Training of Convolutional Neural Network

Convolutional neural network (CNN) [58] is used to estimate the types of states of the cyber system in the partially observable environment. CNN is implemented using Net# language in Microsoft Azure Learning Studio. The network traffic data set is time series data as the data points are made out of successive measurements across the time [92]. Hence, 1-dimensional CNN is used to estimate the types of states of the cyber system using the network traffic data set [93]. In CNN, there are kernels that slide through the dimensions where each kernel describes as a set of weights known as kernel applications. The central nodes are the node of the source layer of CNN which correspond to each kernel application. In Net# program, InputShape represents the dimensionality of the 1-D CNN, KernelShape represents the dimensionality of each kernel for 1-D CNN, Stride denotes the sliding step sizes of CNN, Sharing denotes the weight sharing of each dimension and MapCount denotes the number of feature map for the CNN. The learning rate for the CNN is set to 0.01 and initial weight is set as 0.6 and the number of learning iteration is set as 20.

8.7 Estimation of Types of States of Cyber System

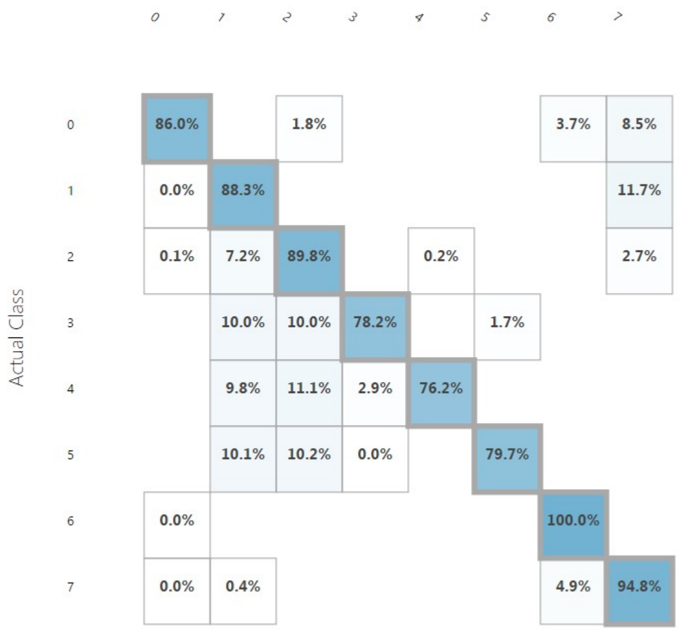
In order to estimate the types of states of the cyber system in the partially observable environment, the testing data sets which are prepared in Section 8.5 are used to test and evaluate the machine learning algorithms. The testing data set is also fed into the trained machine learning algorithms. The testing data set contains 390284 data points. Convolutional neural network classifies each data point of the testing data set and allocates them to different states. The probability of each type of state

Table 16: Estimation of Types of States in Different Partially Observable Environments

Types	**Types of states of the cyber system**							
	State 1	State 2	State 3	State 4	State 5	State 6	State 7	State 8
Type 1	0.0336	0.1371	0.1411	0.0635	0.0219	0.0891	0.2192	0.2941
Type 2	0.0313	0.1416	0.1419	0.0558	0.0194	0.0793	0.2377	0.0292
Type 3	0.0293	0.1424	0.1415	0.0504	0.0172	0.0719	0.2592	0.2890
Type 4	0.0382	0.1197	0.1316	0.0795	0.0273	0.1117	0.2058	0.2858
Type 5	0.0277	0.2056	0.0957	0.0579	0.0198	0.0809	0.2073	0.3045
Type 6	0.0265	0.1593	0.1370	0.0430	0.0148	0.0601	0.2728	0.2862

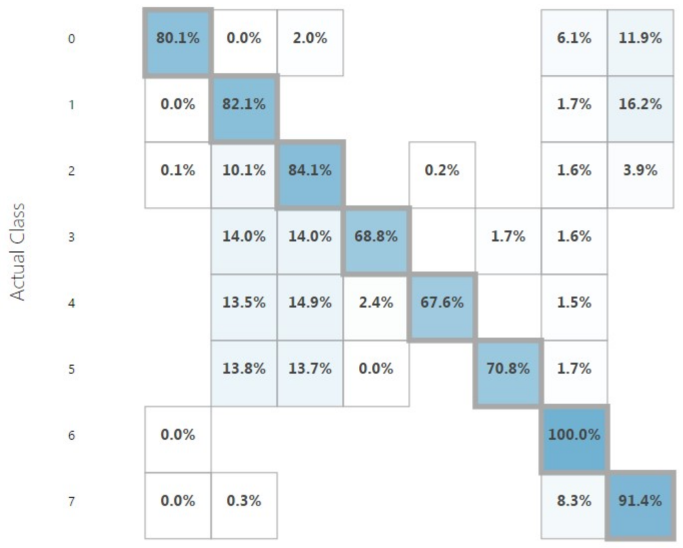
of the cyber system is calculated based on the total number of predicted observations of each type of state of the cyber system and the total number of observations of the testing data set. Table 16 shows the probabilities of eight types of states of the cyber system determined by CNN for six types of partially observable environments. The sum of probabilities of the eight types of states of the cyber system approximate to 1.

Figures 12, 13, 14, 15, 16, 17 shows the confusion matrix for Type 1, Type 2, Type 3, Type 4, Type 5 and Type 6 testing data sets and the confusion matrix describes the performance of the trained classification algorithm.



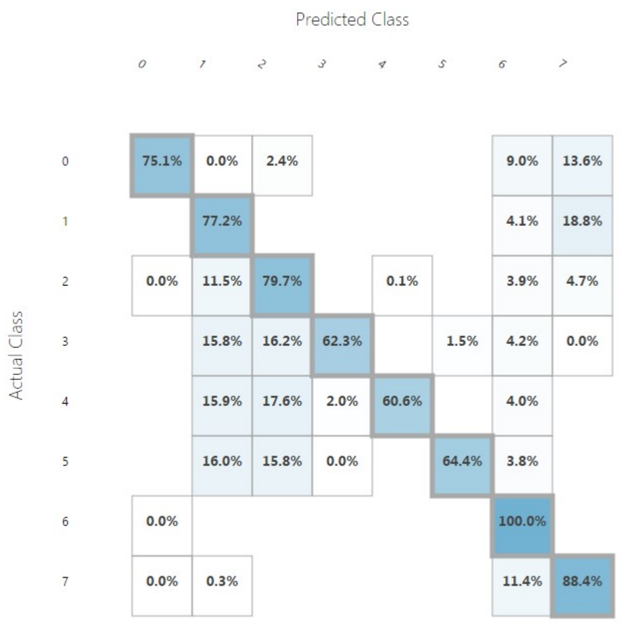
Actual Class	0	1	2	3	4	5	6	7
0	86.0%		1.8%				3.7%	8.5%
1	0.0%	88.3%						11.7%
2	0.1%	7.2%	89.8%		0.2%			2.7%
3		10.0%	10.0%	78.2%		1.7%		
4		9.8%	11.1%	2.9%	76.2%			
5		10.1%	10.2%	0.0%		79.7%		
6	0.0%						100.0%	
7	0.0%	0.4%					4.9%	94.8%

Figure 12: Confusion Matrix in Type 1 partially observable environment



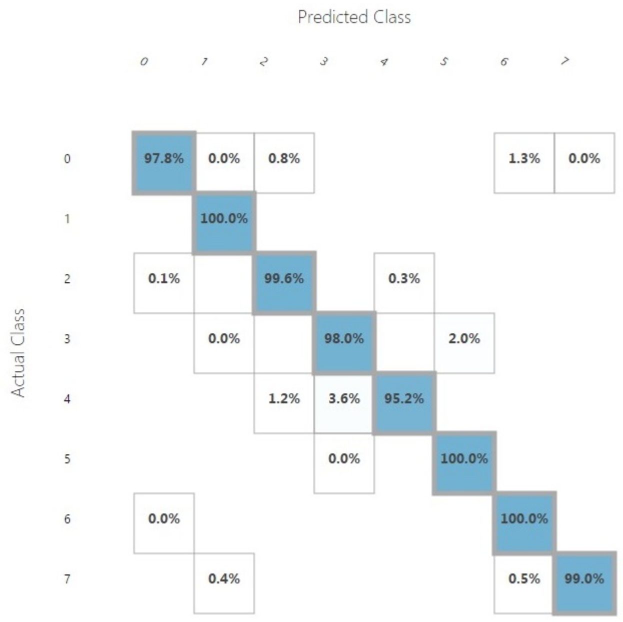
Actual Class								
0	80.1%	0.0%	2.0%				6.1%	11.9%
1	0.0%	82.1%					1.7%	16.2%
2	0.1%	10.1%	84.1%		0.2%		1.6%	3.9%
3		14.0%	14.0%	68.8%		1.7%	1.6%	
4		13.5%	14.9%	2.4%	67.6%		1.5%	
5		13.8%	13.7%	0.0%		70.8%	1.7%	
6	0.0%						100.0%	
7	0.0%	0.3%					8.3%	91.4%

Figure 13: Confusion Matrix in Type 2 partially observable environment


Predicted Class
Actual Class

	0	1	2	3	4	5	6	7
0	75.1%	0.0%	2.4%				9.0%	13.6%
1		77.2%					4.1%	18.8%
2	0.0%	11.5%	79.7%		0.1%		3.9%	4.7%
3		15.8%	16.2%	62.3%		1.5%	4.2%	0.0%
4		15.9%	17.6%	2.0%	60.6%		4.0%	
5		16.0%	15.8%	0.0%		64.4%	3.8%	
6	0.0%						100.0%	
7	0.0%	0.3%					11.4%	88.4%

Figure 14: Confusion Matrix in Type 3 partially observable environment


Predicted Class
Actual Class

	0	1	2	3	4	5	6	7
0	97.8%	0.0%	0.8%				1.3%	0.0%
1		100.0%						
2	0.1%		99.6%		0.3%			
3		0.0%		98.0%		2.0%		
4			1.2%	3.6%	95.2%			
5				0.0%		100.0%		
6	0.0%						100.0%	
7		0.4%					0.5%	99.0%

Figure 15: Confusion Matrix in Type 4 partially observable environment

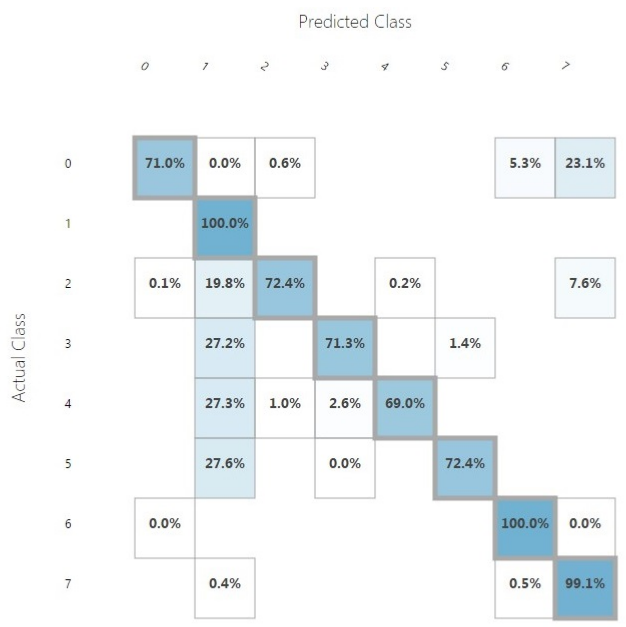


Figure 16: Confusion Matrix in Type 5 partially observable environment

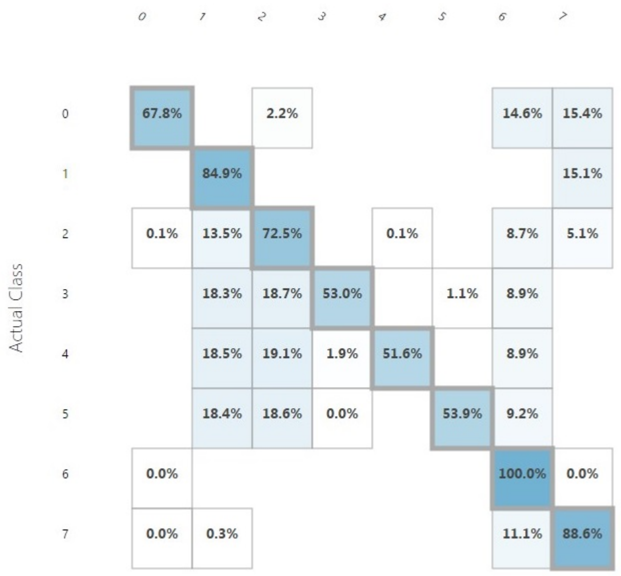


Figure 17: Confusion Matrix in Type 6 partially observable environment

In order to evaluate the state estimation technique for the cyber system in the partially observable environments, the proposed technique is compared with other

Table 17: Micro-averaged precision

	Type 1	**Type 2**	**Type 3**	**Type 4**	**Type 5**	**Type 6**
CNN	90.55	85.88	82.33	99.28	88.75	79.83
MANN	36.83	36.04	35.22	38.12	38.14	34.36
LR	83.27	77.77	74.32	93.97	84.28	77.38
NB	81.10	76.66	73.33	88.60	83.37	78.12
SVM	73.44	69.10	66.10	79.79	72.43	66.21

existing machine learning techniques (Multimodal Artificial Neural Network (ANN) [13], Multimodal Logistic Regression (LR) [48], Multimodal Naïve Bayes (NB) [25] and Multimodal Support Vector Machine (SVM) [26]). Table 17 compares the micro-averaged precision of our trained machine learning algorithm for different types of partially observable environments. Table 18 compares the macro-averaged precision of our trained machine learning algorithm for different types of partially observable environments. Macro-averaged precision provides the average per-class agreement of the types of states of the cyber systems with those of a multimodal classifier. Table 19 compares the micro-averaged recall of our trained machine learning algorithm for different types of partially observable environments. Micro-averaged recall provides the effectiveness of the different classifiers to identify different types of states of the cyber system. Table 20 compares the macro-averaged recall of our trained machine learning algorithm for different types of partially observable environments. Macro-averaged recall provides the average-per class effectiveness of the different classifiers to identify different types of states of the cyber system. Table 21 compares the accuracy of various trained machine learning algorithms for different types of partially observable environments. Accuracy measures the effectiveness of the classifier to identify different types of states of the cyber system.

Table 18: Macro-averaged precision

	Type 1	**Type 2**	**Type 3**	**Type 4**	**Type 5**	**Type 6**
CNN	92.65	89.71	99.13	99.13	93.10	86.56
ANN	NaN	NaN	NaN	NaN	NaN	NaN
LR	81.15	76.99	75.17	94.04	87.61	82.07
NB	63.75	60.28	58.21	71.93	65.15	61.06
SVM	57.20	54.87	53.33	NaN	NaN	52.28

Table 19: Micro-averaged recall

	Type 1	**Type 2**	**Type 3**	**Type 4**	**Type 5**	**Type 6**
CNN	90.55	85.88	82.23	99.28	88.75	79.83
ANN	36.83	36.04	35.22	38.12	38.14	34.36
LR	83.27	77.77	74.32	93.97	84.28	77.38
NB	81.10	76.62	73.33	88.60	83.37	78.12
SVM	73.44	69.10	66.10	79.79	72.43	66.21

Table 20: Macro-averaged recall

	Type 1	**Type 2**	**Type 3**	**Type 4**	**Type 5**	**Type 6**
CNN	86.61	80.60	75.95	98.70	81.91	71.54
ANN	18.36	18.33	18.21	18.18	18.19	18.09
LR	72.98	67.09	63.51	84.50	70.79	62.39
NB	63.89	59.81	56.73	70.18	63.58	57.96
SVM	55.64	50.96	62.56	62.56	50.08	47.03

Table 21: Accuracy

	Type 1	**Type 2**	**Type 3**	**Type 4**	**Type 5**	**Type 6**
CNN	90.55	85.88	82.23	99.28	88.75	79.83
ANN	36.83	36.04	35.32	38.12	38.14	34.36
LR	83.27	77.77	74.32	93.97	84.28	77.38
NB	81.10	76.62	73.33	88.60	83.37	78.12
SVM	73.44	69.10	66.10	79.79	72.43	66.21

Chapter 9

CONCLUSION AND FUTURE RESEARCH

In the thesis, our approach for selecting effective features from the network traffic data sets for detecting attacks in the cyber systems has presented. The reduced number of features of network traffic reduces the computational complexity of training of artificial neural network. Our approach has generated better results for attack detection in cyber systems compared to other existing approaches. We have implemented the attack detection (Step 5) of our approach) as attack detection web service in the cyber systems, which facilitates the use of our approach as Security-as- a-Service (SECaaS) by various resource-constrained devices to detect attacks in cyber infrastructures. We conducted two case studies of our attack detection approach using two comprehensive network traffic data sets. Our attack detection technique is also compared to that of six most recent techniques and our approach has substantial better performance than these existing techniques.

Our attack detection approach has the following limitations. Our attack detection approach is assumed to operate on completely observable cyber systems, it is not applicable to cyber systems which are partially observable. In addition, since MANN can use only supervised machine learning, our approach cannot detect attacks of unknown categories and multistage attacks [94]. The accuracy and efficiency of our approach heavily depend on the quality of the training data sets. In order to improve the accuracy and applicability of our approach to attack detection of large-scale cyber systems, we need to have high-quality training data sets.

In order to have an attack detection approach for partially observable cyber systems, we need to estimate the types of states in partially observable cyber systems. Such an approach is presented involving the use of Elbow method, K-means clustering algorithm and convolutional neural network (CNN). The technique is also compared with other existing machine learning algorithms and shows that CNN provides better accuracy in estimation of the types of states of the cyber systems in various partially observable environments.

It is noted that our approach to estimating the types of states in partially observable environments can be applied in various fields. Besides attack detection in partially observable cyber systems, our approach can also be used for predicting security breaches [3], and change detection [18] in cyber systems, analysis of network behaviors, and network traffic management [27]. Future research need to be done to update the trained machine learning algorithm for estimating the types of states when the cyber operations change continuously.

REFERENCES

[1] Fabio Pasqualetti, Florian Dörfler, and Francesco Bullo. “Attack detection and identification in cyber-physical systems”. In: *IEEE Transactions on Automatic Control* 58.11 (2013), pp. 2715–2729.

[2] Victor Meza, Xiomara Gomez, and Ernesto Perez. “Quantifying observability in state estimation considering network infrastructure failures”. In: *Innovative Smart Grid Technologies Latin America (ISGT LATAM)*. IEEE. 2015, pp. 171–176.

[3] Stephen S Yau, Arun Balaji Buduru, and Vinjith Nagaraja. “Protecting Critical Cloud Infrastructures with Predictive Capability”. In: *IEEE 8th International Conference on Cloud Computing (CLOUD)*. IEEE. 2015, pp. 1119–1124.

[4] Finn V Jensen. *An introduction to Bayesian networks*. Vol. 210. UCL press London, 1996.

[5] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[6] Claudio Mazzariello, Roberto Bifulco, and Roberto Canonico. “Integrating a network ids into an open source cloud computing environment”. In: *Sixth International Conference on Information Assurance and Security (IAS)*. IEEE. 2010, pp. 265–270.

[7] Aman Bakshi and B Yogesh. “Securing cloud from ddos attacks using intrusion detection system in virtual machine”. In: *Second International Conference on Communication Software and Networks*. IEEE. 2010, pp. 260–264.

[8] Chirag Modi et al. “A survey of intrusion detection techniques in cloud”. In: *Journal of Network and Computer Applications* 36.1 (2013), pp. 42–57.

[9] R Sekar et al. “Specification-based anomaly detection: a new approach for detecting network intrusions”. In: *Proceedings of the 9th ACM Conference on Computer and Communications Security*. ACM. 2002, pp. 265–274.

[10] Gregory V Bard and Brian Little. “FLOWHUNT—Specification-Based Intrusion Detection using Neural Networks”. In: (2003).

[11] Kleber Vieira et al. “Intrusion detection for grid and cloud computing”. In: *IT Professional Magazine* 12.4 (2010), pp. 38–43.

[12] Ming-Yang Su, Gwo-Jong Yu, and Chun-Yuen Lin. “A real-time network intrusion detection system for large-scale attacks based on an incremental mining approach”. In: *Computers & Security* 28.5 (2009), pp. 301–309.

[13] Dan W Patterson. *Artificial neural networks: theory and applications*. Prentice Hall PTR, 1998.

[14] David E Goldberg. *Genetic algorithms*. Pearson Education India, 2006.

[15] Huy Anh Nguyen and Deokjai Choi. “Application of data mining to network intrusion detection: classifier selection model”. In: *Challenges for Next Generation Network Operations and Service Management*. Springer, 2008, pp. 399–408.

[16] Nour Moustafa and Jill Slay. “UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)”. In: *Military Communications and Information Systems Conference (MilCIS)*. IEEE. 2015, pp. 1–6.

[17] Mahbod Tavallaee et al. “A detailed analysis of the KDD CUP 99 data set”. In: *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications*. 2009.

[18] Erik Hartikainen and Svante Ekelin. “Enhanced network-state estimation using change detection”. In: *31st IEEE Conference on Local Computer Networks, Proceedings*. IEEE. 2006, pp. 683–693.

[19] Chun-Lien Su and Chan-Nan Lu. “Interconnected network state estimation using randomly delayed measurements”. In: *IEEE Transactions on Power Systems,* 16.4 (2001), pp. 870–878.

[20] David Silver and Joel Veness. “Monte-Carlo planning in large POMDPs”. In: *Advances in Neural Information Processing Systems*. 2010, pp. 2164–2172.

[21] Navjit Singh and Anantdeep Kaur. “A Survey: Multilayer Feed-Forward Neural Network Approaches for Intrusion Detection System”. In: *International Journal For Technological Research In Engineering* 2.11 (2015), pp. 2906–2909.

[22] Nour Moustafa and Jill Slay. “The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set”. In: *Information Security Journal: A Global Perspective* (2016), pp. 1–14.

[23] Ali Shiravi et al. “Toward developing a systematic approach to generate benchmark datasets for intrusion detection”. In: *Computers & Security* 31.3 (2012), pp. 357–374.

[24] Haleh Vafaie and Ibrahim F Imam. “Feature selection methods: genetic algorithms vs. greedy-like search”. In: *Proceedings of International Conference on Fuzzy and Intelligent Control Systems*. 1994, pp. 39–43.

[25] Jingnian Chen et al. “Feature selection for text classification with Naive Bayes”. In: *Expert Systems with Applications* 36.3 (2009), pp. 5432–5435.

[26] Corinna Cortes and Vladimir Vapnik. “Support vector machine”. In: *Machine learning* 20.3 (1995), pp. 273–297.

[27] Moshe Ben-Akiva et al. “Network state estimation and prediction for real-time traffic management”. In: *Networks and Spatial Economics* 1.3-4 (2001), pp. 293–318.

[28] Nicholas Roy, Geoffrey J Gordon, and Sebastian Thrun. “Finding approximate POMDP solutions through belief compression”. In: *J. Artif. Intell. Res.(JAIR)* 23 (2005), pp. 1–40.

[29] Andrew R Liu and Robert R Bitmead. “Stochastic observability in network state estimation and control”. In: *Automatica* 47.1 (2011), pp. 65–78.

[30] Paul Barford and David Plonka. “Characteristics of network traffic flow anomalies”. In: *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*. ACM. 2001, pp. 69–73.

[31] Jack V Tu. “Advantages and disadvantages of using artificial neural networks versus logistic regression for predicting medical outcomes”. In: *Journal of Clinical Epidemiology* 49.11 (1996), pp. 1225–1231.

[32] Avrim L Blum and Pat Langley. “Selection of relevant features and examples in machine learning”. In: *Artificial intelligence* 97.1 (1997), pp. 245–271.

[33] Jinghai Rao and Xiaomeng Su. “A survey of automated web service composition methods”. In: *Semantic Web Services and Web Process Composition*. Springer, 2004, pp. 43–54.

[34] Felix Fuentes and Dulal C Kar. “Ethereal vs. Tcpdump: a comparative study on packet sniffing tools for educational purpose”. In: *Journal of Computing Sciences in Colleges* 20.4 (2005), pp. 169–176.

[35] Richard Blum. *Network Performance Open Source Toolkit: Using Netperf, tcptrace, NISTnet, and SSFNet.* John Wiley & Sons, 2003.

[36] Andrew W Moore. "Cross-validation for detecting and preventing overfitting". In: *School of Computer Science Carnegie Mellon University* (2001).

[37] Alan Saied. "Distributed Denial of Service (DDoS) Attack Detection and Mitigation". PhD thesis. King's College London, 2015.

[38] Angela Orebaugh, Gilbert Ramirez, and Jay Beale. *Wireshark & Ethereal network protocol analyzer toolkit.* Syngress, 2006.

[39] Hans Braunschweiler. "Security as a Service". In: *Geneva Papers on Risk and Insurance–Issues and Practice* 3.7 (1978), pp. 42–55.

[40] Mujahid Tabassum and Kuruvilla Mathew. "A genetic algorithm analysis towards optimization solutions". In: *International Journal of Digital Information and Wireless Communications (IJDIWC)* 4.1 (2014), pp. 124–142.

[41] Kuo-Hsiu Chen, Hong-Ling Chen, and Hahn-Ming Lee. "A multiclass neural network classifier with fuzzy teaching inputs". In: *Fuzzy Sets and Systems* 91.1 (1997), pp. 15–35.

[42] Sepp Hochreiter. "The vanishing gradient problem during learning recurrent neural nets and problem solutions". In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6.02 (1998), pp. 107–116.

[43] Włodzisław Duch and Norbert Jankowski. "Survey of neural transfer functions". In: *Neural Computing Surveys* 2.1 (1999), pp. 163–212.

[44] Michael J Berry and Gordon Linoff. *Data Mining Techniques: For Marketing, Sales, and Customer Support.* John Wiley & Sons, Inc., 1997.

[45] ATC Goh. "Back-propagation neural networks for modeling complex systems". In: *Artificial Intelligence in Engineering* 9.3 (1995), pp. 143–151.

[46] Xinyou Yin et al. "A flexible sigmoid function of determinate growth". In: *Annals of Botany* 91.3 (2003), pp. 361–371.

[47] Ting-Hua Yi, Hong-Nan Li, and Xiao-Yan Zhao. "Noise smoothing for structural vibration test signals using an improved wavelet thresholding technique". In: *Sensors* 12.8 (2012), pp. 11205–11220.

[48] David W Hosmer Jr and Stanley Lemeshow. *Applied logistic regression.* John Wiley & Sons, 2004.

[49] P Natesan and P Balasubramanie. "Multi stage filter using enhanced adaboost for network intrusion detection". In: *International Journal of Network Security & Its Applications* 4.3 (2012), pp. 121–135.

[50] James Newsome et al. "Dynamic taint analysis: Automatic detection, analysis, and signature generation of exploit attacks on commodity software". In: *Proceedings of the 12th Network and Distributed Systems Security Symposium.* 2005.

[51] Andrew S Tanenbaum. "Computer networks, 4-th edition". In: *ed: Prentice Hall* (2003).

[52] Fazel Famili et al. "Data pre-processing and intelligent data analysis". In: *International Journal on Intelligent Data Analysis* 1.1 (1997).

[53] Pratik Narang, Vansh Khurana, and Chittaranjan Hota. "Machine-learning approaches for P2P botnet detection using signal-processing techniques". In: *Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems.* ACM. 2014, pp. 338–341.

[54] Pabitra Mitra, CA Murthy, and Sankar K Pal. "Unsupervised feature selection using feature similarity". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.3 (2002), pp. 301–312.

[55] Trupti M Kodinariya and Prashant R Makwana. "Review on determining number of Cluster in K-Means Clustering". In: *International Journal* 1.6 (2013), pp. 90–95.

[56] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. "Unsupervised learning". In: *The Elements of Statistical Learning.* Springer, 2009, pp. 485–585.

[57] Chris Ding and Xiaofeng He. "K-means clustering via principal component analysis". In: *Proceedings of the Twenty-First International Conference on Machine Learning.* ACM. 2004, pp. 29–37.

[58] Yoon Kim. "Convolutional neural networks for sentence classification". In: *arXiv preprint arXiv:1408.5882* (2014).

[59] Matthew D Zeiler and Rob Fergus. “Visualizing and understanding convolutional networks”. In: *European Conference on Computer Vision*. Springer. 2014, pp. 818–833.

[60] Deng Cai, Chiyuan Zhang, and Xiaofei He. “Unsupervised feature selection for multi-cluster data”. In: *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM. 2010, pp. 333–342.

[61] Paul Barford et al. “A signal analysis of network traffic anomalies”. In: *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*. ACM. 2002, pp. 71–82.

[62] Elaheh Biglar Beigi et al. “Towards effective feature selection in machine learning-based botnet detection approaches”. In: *IEEE Conference on Communications and Network Security (CNS)*. IEEE. 2014, pp. 247–255.

[63] Istvin Csabai. “1/f noise in computer network traffic”. In: *Journal of Physics A: Mathematical and General* 27.12 (1994), pp. L417–L421.

[64] Erhard Rahm and Hong Hai Do. “Data cleaning: Problems and current approaches”. In: *IEEE Data Eng. Bull.* 23.4 (2000), pp. 3–13.

[65] Lei Yu and Huan Liu. “Feature selection for high-dimensional data: A fast correlation-based filter solution”. In: *ICML*. Vol. 3. 2003, pp. 856–863.

[66] Isabelle Guyon and André Elisseeff. “An introduction to feature extraction”. In: *Feature extraction*. Springer, 2006, pp. 1–25.

[67] Charles Rader. “Discrete Fourier transforms when the number of data samples is prime”. In: *Proceedings of the IEEE* 56.6 (1968), pp. 1107–1108.

[68] Mian Zhou and Sheau-Dong Lang. “Mining frequency content of network traffic for intrusion detection”. In: *Proceedings of the IASTED International Conference on Communication, Network, and Information Security*. 2003, pp. 101–107.

[69] Peter MG Apers, Henk M Blanken, and Maurice AW Houtsma. *Multimedia database in perspective*. Springer Science & Business Media, 2012.

[70] Bernhard Tellenbach et al. “Beyond shannon: Characterizing internet traffic with generalized entropy metrics”. In: *Passive and Active Network Measurement*. Springer, 2009, pp. 239–248.

[71] Claude E Shannon et al. “Two-way communication channels”. In: *Proc. 4th Berkeley Symp. Math. Stat. Prob.* Vol. 1. Citeseer. 1961, pp. 611–644.

[72] Jundong Li et al. "Feature Selection: A Data Perspective". In: *arXiv preprint arXiv:1601.07996* (2016).

[73] Jiliang Tang et al. "Discriminant Analysis for Unsupervised Feature Selection." In: *SDM*. SIAM. 2014, pp. 938–946.

[74] Sara Laviosa. *Corpus-based Translation Studies: Theory, Findings, Applications*. Vol. 17. Rodopi, 2002.

[75] Purnima Bholowalia and Arvind Kumar. "EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN". In: *International Journal of Computer Applications* 105.9 (2014).

[76] Osama Abu Abbas et al. "Comparisons Between Data Clustering Algorithms". In: *Int. Arab J. Inf. Technol.* 5.3 (2008), pp. 320–325.

[77] Min Lin, Qiang Chen, and Shuicheng Yan. "Network in network". In: *arXiv preprint arXiv:1312.4400* (2013).

[78] Jiquan Ngiam et al. "Tiled convolutional neural networks". In: *Advances in Neural Information Processing Systems*. 2010, pp. 1279–1287.

[79] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems*. 2012, pp. 1097–1105.

[80] Dan C Ciresan et al. "Flexible, high performance convolutional neural networks for image classification". In: *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*. Vol. 22. 1. 2011, pp. 1237–1242.

[81] Serkan Kiranyaz, Turker Ince, and Moncef Gabbouj. "Real-Time Patient-Specific ECG Classification by 1-D Convolutional Neural Networks". In: *IEEE Transactions on Biomedical Engineering* 63.3 (2016), pp. 664–675.

[82] Yann LeCun and Yoshua Bengio. "The Handbook of Brain Theory and Neural Networks". In: ed. by Michael A. Arbib. Cambridge, MA, USA: MIT Press, 1998. Chap. Convolutional Networks for Images, Speech, and Time Series, pp. 255–258. ISBN: 0-262-51102-9.

[83] Guillaume Bouchard. "Efficient bounds for the softmax function and applications to approximate inference in hybrid models". In: *NIPS 2007 Workshop for Approximate Bayesian Inference in Continuous/Hybrid Systems*. Citeseer. 2007.

[84] Fabian Pedregosa et al. "Scikit-learn: Machine learning in Python". In: *The Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[85] Keith R Jackson et al. "Performance analysis of high performance computing applications on the amazon web services cloud". In: *IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom), 2010*. IEEE. 2010, pp. 159–168.

[86] Borja Merino. *Instant Traffic Analysis with Tshark How-to*. Packt Publishing Ltd, 2013.

[87] Charles Rader. "Discrete Fourier transforms when the number of data samples is prime". In: *Proceedings of the IEEE* 56.6 (1968), pp. 1107–1108.

[88] Dave Shreiner, Mason Woo, and Jackie Neider. "OpenGL (R) Programming Guide: The Official Guide to Learning OpenGL, Version 1.2". In: (2008).

[89] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[90] Travis E Oliphant. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[91] Eric Jones, Travis Oliphant, P Peterson, et al. *Open source scientific tools for Python*. 2001.

[92] Manish Joshi and Theyazn Hassn Hadi. "A Review of Network Traffic Analysis and Prediction Techniques". In: *arXiv preprint arXiv:1507.05722* (2015).

[93] Yi Zheng et al. "Time series classification using multi-channels deep convolutional neural networks". In: *Web-Age Information Management*. Springer, 2014, pp. 298–310.

[94] Shanchieh J Yang et al. "High level information fusion for tracking and projection of multistage cyber attacks". In: *Information Fusion* 10.1 (2009), pp. 107–121.