

# **Cognitive Unsupervised Clustering for Detecting Cyber Attacks**

By

Kaiser Nahiyani

A Thesis submitted to the Faculty of Graduate Studies of

The University of Manitoba

in partial fulfillment of the requirements of the degree of

**MASTER OF SCIENCE**

Department of Electrical and Computer Engineering

University of Manitoba, Winnipeg, MB, Canada

Copyright © 2020 by Kaiser Nahiyani

## **Abstract**

*It has always been a challenge to extract meaning out of unstructured data. In the field of network intrusion detection, the availability of structured, labelled datasets is limited. Most approaches adhere to techniques that demand high-end computing resources, and do not yield satisfactory results; hence human analysts must examine all the network events in order to isolate intrusion attempts. This study proposes an intelligent approach of extracting information out of large unstructured and unlabeled datasets and performs unsupervised detection of attack traffic from normal network traffic, utilizing the concepts of cognitive learning, complexity analysis, and statistical higher-order feature learning. The thesis aims to develop a methodology for the human analysts to disregard a major portion of the network dataset that contains regular traffic, and focus on the finite time-windows that have been subjected to potential attacks. Statistical higher-order feature extraction from network flows was used to create significant features out of the large unlabelled network intrusion detection dataset, which was later classified using unsupervised kmeans clustering and variance fractal dimension trajectory (VFDT) based complexity analysis. The proposed methodology has been validated using the UNSW-NB15 network intrusion dataset and the performance measures used are; detection accuracy, false positive and false negative rate, Receiver Operation Characteristics curve, Area Under Curve Value, and F1 score. Subsequently, a comparative analysis of the proposed model with a prominent traditional unsupervised machine learning technique (i.e. standard kmeans clustering) based scheme has been performed to evaluate and benchmark the efficacy of the proposed methodology. The empirically validated results show that the proposed cognitive unsupervised clustering technique-based model outperforms the general unsupervised detection scheme based on performance measures such as detection accuracy, false positive and false negative rates, Area Under Curve Value and F1 score.*

## **Acknowledgements**

First and foremost, I am very thankful to the Almighty ALLAH (SWT) for everything that He has bestowed upon me. I am ever obliged to Him for giving me the interest to pursue graduate degrees and for giving me the strength to continue till completion.

It is with sincere gratitude that I thank my supervisor Prof. Dr. Ken Ferens for supporting me throughout my journey as a graduate student at University of Manitoba. Without his cooperation, guidance, advice and motivation I would not have been the person that I am today. I am blessed to have worked with such an amazing teacher, and a passionate mentor.

I would like to thank Prof. Dr. Bob McLeod, his interest in my research and encouragement have always motivated me deeply. I would also like to appreciate Prof. Dr. Noman Mohammed for their interest in my research and for the valuable time given in evaluating my thesis.

I sincerely thank my beloved wife for being my support system through every thick and thin. I am grateful to my parents, siblings, in-laws and friends for their love and support. I am forever indebted to my parents for their sacrifices and their encouragements. I am thankful to my elder sister who have supported me a lot throughout my life, and took great care of things back home while I was away pursuing my higher studies.

## **Dedication**

I dedicate this work to my loving parents, who have been teachers themselves. Thank you for ingraining the passion for seeking knowledge in me. Everything that I have achieved in my life is because of the sacrifices that you have made.

# Table of Contents

<b>Abstract .....</b>	<b>ii</b>
<b>Acknowledgements.....</b>	<b>iii</b>
<b>Dedication .....</b>	<b>iv</b>
<b>List of Figures .....</b>	<b>vii</b>
<b>List of Tables.....</b>	<b>ix</b>
<b>Chapter 1 Introduction.....</b>	<b>1</b>
1.1 Thesis Statement and Overview .....	4
1.2 Outline of the Thesis .....	5
<b>Chapter 2 Background Study .....</b>	<b>6</b>
2.1 Intrusion Detection Overview .....	6
2.1.1 Intrusion Detection System .....	6
2.1.2 Types of Intrusion Detection System .....	7
2.1.3 Signature based and Anomaly based IDS.....	10
2.1.4 Related work on Unsupervised Anomaly Based Intrusion Detection .....	12
2.2 Statistical Learning.....	13
2.3 Cognitive Cyber Informatics and Complexity Analysis using VFDT .....	17
<b>Chapter 3 Proposed Algorithm.....</b>	<b>22</b>
3.1 Data Cleaning .....	22
3.2 Missing Data .....	24
3.3 Data Encoding .....	25
3.4 Feature Engineering .....	26
3.5 Sampling of the Network Dataset .....	27
3.6 Feature Extraction .....	30
3.7 Unsupervised Classification Using K-means Clustering .....	35
3.8 Attack Detection.....	36
3.9 Proposed Process Model .....	39

3.10 Proposed Network Topological Model .....	40
3.11 Workflow of Proposed Algorithm.....	43
<b>Chapter 4 Experiments and Results .....</b>	<b>45</b>
4.1 System Setup .....	45
4.2 Performance Evaluation .....	45
4.3 Experimental Dataset .....	46
4.4 Experiment 1: Exploring the dataset .....	47
4.5 Experiment 2: Applying a basic unsupervised clustering model on the dataset .....	58
4.6 Experiment 3: Execution of the Proposed Algorithm .....	58
4.7 Results .....	59
<b>Chapter 5 Conclusion and Future Work .....</b>	<b>64</b>
<b>References .....</b>	<b>68</b>
<b>APPENDIX A - List of the features of UNSW-NB15 Dataset.....</b>	<b>76</b>

## List of Figures

Figure 1 Network-based Intrusion Detection	8
Figure 2 Host-based Intrusion Detection	9
Figure 3 Data Cleaning - getting rid of anomalies	23
Figure 4 Script to replace irregularities from the file	23
Figure 5 Missing Data - UNSW	24
Figure 6 Sampler and Feature Extractor	28
Figure 7 time-windows Elaborated	29
Figure 8 Feature Extraction Elaborated	31
Figure 9 Sampler and Feature Extractor [71]	36
Figure 10 A 2D example of two separate cluster groups	37
Figure 11 Attack Isolation Explained	39
Figure 12 Proposed Model	40
Figure 13 Network Topology Explained	42
Figure 14 Flowchart of Proposed Method	43
Figure 15 Attack and Normal Counts for UNSW Dataset	47
Figure 16 Histogram Analysis Of Fields 'Dintpkt', 'Djit', 'Sintpkt', 'Sjit', 'ct_dst_ltm', 'ct_dst_sport_ltm', 'ct_srv_dst', 'ct_srv_src'	48
Figure 17 Histograms for features Dload, Dpkts, Sload, Spkts, ct_dst_src_ltm, ct_flw_http_mthd, ct_state_ttl, dbytes	49
Figure 18 Histograms for features ct_src_dport_ltm, ct_src_ltm, dloss, dmeanz, is_ftp_login, is_sm_ips_ports, stcpb, sttl	50
Figure 19 Histograms for features dur, dwin, dtcp, dttl, res_bdy_len, sbytes, swin, synack	51
Figure 20 Histograms for features sloss, smeanz, tcprtt, trans_depth	52
Figure 21 Value Count Chart - Service	53
Figure 22 Value Count Chart - ct_srv_dst	53
Figure 23 Value Count Chart - Proto	54

<i>Figure 24 Value Counts – SRCIP (UNSW-NB15)</i>	55
<i>Figure 25 Count of events per Second (UNSW-NB15 file 1)</i>	55
<i>Figure 26 Scatter and Density Plot (UNSW-NB15) [72]</i>	57
<i>Figure 27 Comparison of No of Dataset Rows To Process</i>	59
<i>Figure 28 Comparison of Execution Time</i>	60
<i>Figure 29 ROC Curve of the Proposed Algorithm</i>	62
<i>Figure 30 Confusion matrix - proposed cognitive unsupervised model</i>	63



**List of Tables**

*Table 1 List of Categorical Attributes- UNSW* \_\_\_\_\_ 26

*Table 2 Network Communication* \_\_\_\_\_ 41

*Table 3 Results Obtained* \_\_\_\_\_ 61

## Chapter 1 Introduction

In the modern era, use of technology and online resources is an absolutely necessity. The use of search engines, emails, online portals, ecommerce websites, streaming services, access to vast number of utilities and services over the internet has incrementally made this virtual world into a huge complex mesh [1]. These online entities, with all the information that they withhold, are of vital significance and needs to be secured against the evildoers. Securing these cyber assets needs robust monitoring, precise preventive defense mechanisms and, accurate and timely detection of cyber-attack and intrusion attempts. However, detecting these isolated sporadic attacks amidst the huge loads of normal traffic is a tough task. The complex infrastructure and ever-growing network traffic often make it impossible for the limited number of system admins to manage the network efficiently. Inspecting all the network elements and investigating through all the normal traffic to isolate the threats exploited in real time is a tremendously exhaustive process. That is why supplementary automated prevention and monitoring systems are required to help the human workforce in detecting intrusion attempts. However, crafting these algorithms that, can outperform the ever-changing intrusion attempts, and are able to detect the zero-day attacks are very difficult to construct and need more research focus. For the data scientists, working with unlabeled data is one of the most challenging tasks. Without any labelled data sources, it is not possible to train the model with standard training data. Supervised learning models are easier to design as there are available labelled datasets to train the models. Supervised learning models are fed with labelled data repeatedly, and in every epoch the model is fed with the entire dataset. In each iteration of learning the dataset, the

model incrementally aligns itself to the data presented, and over the course of repetitive learning, it becomes more and more efficient in categorizing items during the testing phase. However, supervised models are not good with never-seen-before attacks, and having the attackers always working on novel approaches to breach the security, it is necessary to put in place some checks that are able to filter out anomalies that are new. Hence unsupervised detection is quite a desirable field of study and research in cybersecurity but attaining high detection accuracy is often not possible. Hence, for these unsupervised algorithms, accuracy is often not the motive, rather the focus is towards isolating a small group of most-probable attacks, or, at the least the odd events including some false positives, from the vast group of regular homogeneous network traffic.

The other significance of unsupervised learning is to make use of the unlabeled data, and to skip the tedious process of data labelling. Data is getting generated in millions of Peta Bytes every second and capturing and preserving colossal amounts of data in an accurately labelled state is often expensive and economically infeasible. The captured raw data needs to go through levels of processing before they can be deemed as effective datasets containing orderly information suitable for use in machine learning models. For network data, which is usually packet captures of the raw data bytes flowing through the network, the raw data can provide little to no knowledge until they are read through special software like Wireshark. Therefore, solving challenging problems need efficient algorithms that will learn from unlabeled datasets by extracting meaning out of data, without knowing which class it represents. This is particularly important in the field of cybersecurity where extracting meaning out of packet captures is hard and converting packet captures into labelled datasets is a complicated and resource-hungry process.

The latest cyber threat detection platforms utilize either the perceivable known signatures or heuristic-based behavioral analysis to determine the presence of potential threats. Therefore, it is imperative that the human traits and capabilities are an innate attribute required to analyze and differentiate the true positives from the large pool of normal traffic, and hence this implies that we need to improve the cyber defense tools to integrate human-like cognition capabilities. This claims further credit since humans are the perfect learners, especially when it comes into learning new things. Therefore, in this work, we are incorporating cognitive approach to isolate the attack from the normal traffic.

The traditional machine learning algorithms are performing single-scale analysis. The single-scale approaches used in analyzing autonomous intelligent systems and natural cognitive processes are good for all patterns and processes that are scale independent or single-scale in nature. Such approaches are not adequate for the systems that are inherently scale-free, like network traffic[2]. Hence if a multi-scale approach is required to capture the local characteristic of the system that contains the information of any underlying complex behavior. The proposed solution will be using multiscale complexity analysis using variance fractal dimension. The unsupervised nature of the algorithm enables it to be effective not only for known attacks but also for unknown 0-day attacks.

Focusing on the key motivation of cognition-inspired adaptive learning, this study also tries to adopt significant properties of early-stage human learning. It has been studied that the ability of humans to identify and recognize objects develop shortly after birth [3] [4]. A very significant skill that they develop during this time to supports object perception is gaze control, that is, the ability to direct gaze toward informative or distinctive regions of an object, like edges and contours, as well as to shift gaze from one part of the object to

another [5] [6] [7]. While focusing on the regions, one of the key processes that they use to identify distinctive and informative regions is by using statistical learning, which is a vital process that infants use to assess their surroundings [8]. Hence the perfect approach to imitating human-learning is segment-wise focus, extraction of statistical features, with complexity analysis as the final layer of discriminative analysis, which are the exact building blocks for the proposed algorithm.

## **1.1 Thesis Statement and Overview**

The thesis presents a cognitive unsupervised clustering algorithm for intrusion detection that extracts significant features out of an unlabeled dataset, and separates the cluster representing normal traffic from the cluster representing attack traffic, maintaining a reasonable success rate. For testing the algorithm two different datasets will be used - UNSW-NB15 dataset (Australian Centre for Cyber Security) [9], [10] which is considered as a very reliable and extensively worked upon dataset. The specific research questions that we will be trying to resolve are listed below.

1. Is it possible to use complexity analysis to differentiate clusters of normal packets and attack packets, in other words, can a complexity measure obtain different results for an attack cluster and a normal cluster?
2. Is it possible to use unsupervised learning to obtain separable clusters of normal and attack packets?
3. Can a network dataset be substantially reduced and still hold enough information to apply clustering techniques to separate attacks from normals?

## 1.2 Outline of the Thesis

This thesis is structured in following chapters:

- |                               |  |
|-------------------------------|--|
| 1. Introduction               | In this section, the problem is introduced, the specific research questions are stated, and the organization of the thesis is elaborated.                            |
| 2. Background Study           | We review relevant aspects of intrusion detection systems and lay the theoretical background necessary for the readers to understand the methodology proposed later. |
| 3. Proposed Algorithm         | We present the dataset, the features extraction process and proposed the algorithm in detail.  |
| 4. Experiments and Results    | We present the experimentation setup and the result here in this section.  |
| 5. Conclusion and Future Work | Here we discuss how well the goals of the thesis were achieved and avenues of future research.   |

## **Chapter 2 Background Study**

The domain of intrusion detection and prevention systems have been explored for years and there has been quite a lot of research focused in the field of cognitive cyber informatics. In this chapter, we will gradually elaborate on the various concepts required to understand our proposed algorithm. Firstly, we need to emphasize on intrusion detection systems and understand the various types on IDS. Further, we focus on some notable previous research works on traditional Unsupervised Anomaly Based Intrusion Detection techniques. Later we turn our focus into some cognitive cyber informatics and complexity measure analysis using fractal measures. On the last section we elaborate the concepts related to K-Means Clustering. The concepts learned from each of the components will act as building blocks towards the proposed algorithm.

### **2.1 Intrusion Detection Overview**

Intrusion detection systems, popularly known as IDS is an important component in the modern day network defense framework. Let us focus further on intrusion detection systems, and the broad categories of intrusion detection systems.

#### **2.1.1 Intrusion Detection System**

Intrusion detection system is the setup that captures violations of network security, it collects key information from network and systems to analyze further and detect occurrences of any events or actions that undermines the security strategy for an

environment. IDS can proactively monitor network and alert that system admins before the harm is commenced and are able to mitigate the extent of the damage incurred to the secured resources [1]. The system examines intrusion attempts before the resources gets damaged and uses various alert mechanisms to circulate communication of the potential threat to appropriate stakeholders to allow necessary preventive measures to be taken promptly. With an intrusion detection system in place, incidents of repeated violation through the same mechanism can be avoided.

### **2.1.2 Types of Intrusion Detection System**

Intrusion detections systems are based on two categories, depending on where the system operates.

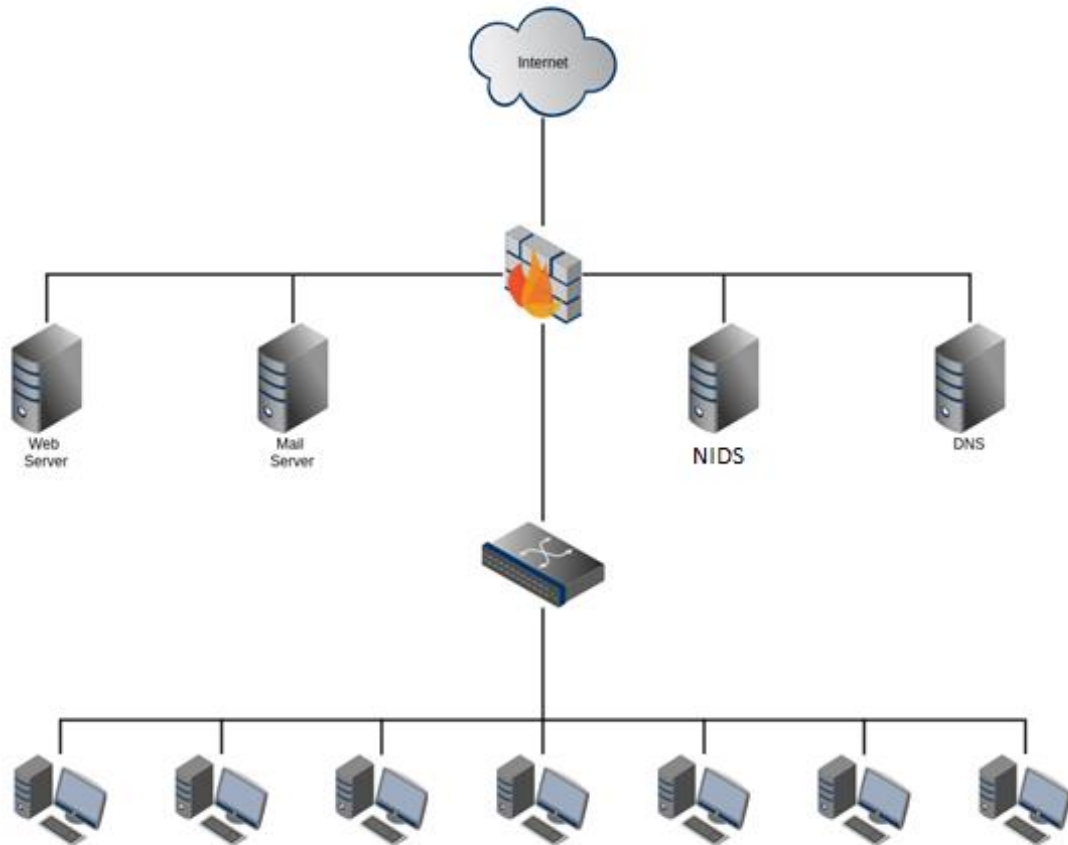
- i. Network Based Intrusion Detection System
- ii. Host Based Intrusion Detection System
- iii. Hybrid Based Intrusion Detection System

#### **i. Network Based Intrusion Detection System**

In network-based intrusion detection systems, the primary information on intrusion is gained by monitoring the inflow and outflow of packets in the network. This analysis of the network traffic is done by connecting to the main hub, router and the switches in the network, and then monitoring the traffic to and from the various hosts, and by doing port monitoring or network tapping. An example of NIDS is Snort. They are the most popular



types of intrusion detection systems and act as a passive network monitoring tool that keeps monitoring the traffic for malicious or unwanted events.

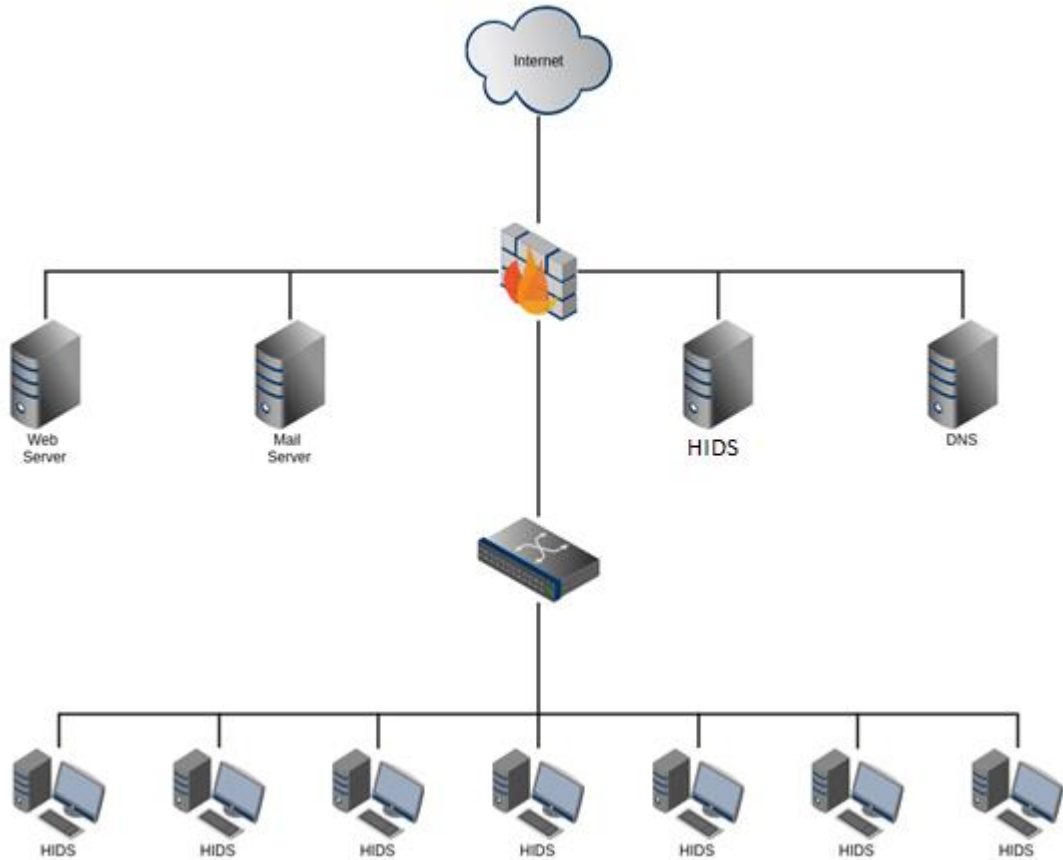


**FIGURE 1: NETWORK-BASED INTRUSION DETECTION**

## **ii. Host Based Intrusion Detection System**

In host based intrusion detection systems, there is a component of the intrusion detection system built into the hosts (servers) themselves, which monitors the activity of each and every host in the network, and reports to a central server not only the incoming and outgoing traffic but takes into account their process tree and other system parameters that depict the status of the host and the tasks that it is performing. The system monitors various

elements like identification and authentication mechanisms, system calls, login and logout attempts, files open and other user activity [12].



**FIGURE 2: HOST-BASED INTRUSION DETECTION**

### **iii. Hybrid Based Intrusion Detection System**

In Hybrid based IDS, both the host and the network-based IDS principles are put into play. Not only are there monitoring on the network traffic itself, there also are primary information on intrusion in gained by monitoring the inflow and outflow of packets in the

network. This analysis of the network traffic is done by connecting to the main hub, router and the switches in the network, and then monitoring the traffic to and from the various hosts, and by doing port monitoring or network tapping. An example of NIDS is Snort. They are the most popular types of intrusion detection systems and act as a passive network monitoring tool that keeps monitoring the traffic for malicious or unwanted events. [12].

### **2.1.3 Signature based and Anomaly based IDS**

Intrusion detection has been broadly categorized into signature based and anomaly-based detection [12].

**Signature based** systems identify intrusion by looking into some predefined patterns associated with attacks. An alarm is set out if these specific behavior or pattern emerge in the system, and optionally it triggers a set of predefined tasks to secure the network from damage [13] [14]. These patterns are preset by human analysts from a-priori analysis, hence in other words, there is a certain level of human-knowledge and cognition incorporated into the signature-based systems. This is one of the reasons why these systems are quite efficient in detecting the previously known attack types. However, the attack landscape is ever-changing and novel attacks emerge, which can't be restricted by these systems as their attack signatures are never encountered before.

The other type of IDS is **anomaly-based intrusion detection systems** [14], these systems map the usual traffic pattern, and rely on the ideology that attacks have traffic pattern that are significantly different from the usual traffic pattern. Anomaly or outlier-based intrusion detection systems are further categorized based on the learning mechanism.

**Supervised-based anomaly detection techniques** are dependent on labelled datasets which are gathered from beforehand, containing both normal traffic and unusual traffic, and once the model has completely learned, it can identify if the unlabeled traffic pattern is a deviation from normal. [15]. However, the usual traffic has more normal traffic than the attack traffic and hence most available real dataset are unbalanced, which impacts the overall efficiency of the model. When the model learns from such a dataset, more of the knowledge within the model is coming from the normal traffic and less from the abnormal or attack traffic which results into less efficient recall. **Semi-supervised anomaly detection techniques**, on the contrary, do not require a two-class dataset, and can be trained using either entirely with attack traffic or entirely normal traffic [16]. This allows the model to be learned entirely with the normal traffic which is abundant in source. However, the disadvantage is that the model learned using semi-supervised techniques are unable to recognize all the anomalies as many anomalies tend to match the normal traffic to disguise and deliver.

Since any form of supervised or semi-supervised learning requires training datasets, this limits the application of the process to certain areas. For the approach to be generic and applicable to any domain, we must apply **Unsupervised Anomaly Detection Technique**. Unsupervised anomaly-based technique doesn't rely on a labelled dataset for training, rather it determines two separate groups within the data it is presented with, in the case of intrusion detection, one being the normal data and the other being intrusion traffic [12]. Unsupervised anomaly detection works on the underlying concept that the normal data traffic has similarities which enables them to be collocated, and that the intrusion attempts

make a significant different pattern from the normal traffic which make them outliers.

#### **2.1.4 Related work on Unsupervised Anomaly Based Intrusion Detection**

In [17], the authors try to compare the application of machine learning in cybersecurity to the other areas where machine learning is more successful usually. They claim that intrusion detection is fundamentally different from other applications of machine learning which makes it significantly difficult to construct such machine learning intrusion detection models into 'real world' operational settings.

The authors in [18] designed an intrusion detection system based on Random Forest algorithm and used KDD'99 dataset for evaluation. In their work they provided three data-mining based frameworks and apply their algorithm on each approach and evaluate their results. In [19], three different types of approaches for intrusion classification was proposed: computation based, artificial intelligence based and biological concepts based, which in [20], the authors claim the abovementioned study provides insufficient detail to the detection approach, and they propose five subcategories based on detection approach, and they categorize all previous literature in this filed into their proposed subcategories and also assess pros and cons of each of the approaches. As the application of machine learning in detecting malicious activities and intrusion attempts grew in popularity, so did the application of unsupervised learning techniques in detecting intrusion attempts and abnormalities [21]. In [22], the authors provided a density and grid based adaptive clustering technique on KDD Cup 1999 dataset, it produced good accuracy but suffered from a high false positive rate. Later in [23], where the authors implemented Genetic

Algorithm and hierarchical clustering-based classifier, the results were better. The authors in [24] applied K-Means algorithm in intrusion detection and achieved very promising results on a subset from KDD-99 dataset. In [25] K-Means clustering has been used with supervised tree models to determine anomalies in the network. K-Means algorithm was again used in [26], where the performance of unsupervised detection on large network intrusion detection datasets was improved by implementing mini-batch K-Means algorithm with PCA used for feature extraction.

Clustering has been a popular unsupervised data exploratory technique for years, and recently, it has emerged as an anomaly detection technique [27] [28]. Clustering, in other words, the technique of partitioning the set of unlabeled data into patterns or clusters [29], this process can be further enhanced with the use of feature learning or feature selection [30]. When new features are created out of the original data, the data is enhanced with new expressions that can make it more suitable for the machine learning model to interpret. With feature transformation techniques, original features are transformed into new forms which enables the machine learning model to absorb the underlying pattern in the data without requiring to learn unnecessary features [31].

## **2.2 Statistical Learning**

Our approach is highly influenced by cognitive learning, and studies show that statistical learning is an important fundamental core of human learning. The first portion of this sections provides references of the abovementioned claim, and the second portion

presents some work of statistical features being used in the field of network traffic analysis and cybersecurity network intrusion detection systems.

Intuitive statistical abilities of young infants have been examined in physical reasoning [32][33], word and scene segmentation [34][8], language learning [35][36][37], and causal reasoning [38][39][40]. Some of the related research will be discussed below.

In [8], the study shows that children as young as in their 8 months can accomplish word segmentation, which is a fundamental tool required for language acquisition solely by statistical relationships between neighboring speech sounds. Furthermore, the process took place with just 2 minutes of exposure, hence the infants must have very powerful mechanism for the computation of statistical properties of the language input.

In [41], authors explore natural image statistics and provide a computational modeling of biological visual systems where they propose a statistical-ecological model and provide predictions largely validated by classic neuroscientific measurements. There they claim that the images that our eyes receive has certain statistical properties (regularities) and that the visual system learns a model of these statistical properties. And this statistical learning from the external systems forms an important factor of inferencing from the surrounding environment.

For years, authors have explored the significance of statistical learning in human learning, memory, intelligence and inductive inference. Major contributors in the field of cognitive science, from Helmholtz[42], Mach[43], and Pearson[44], and continuing through Craik[45], Tolman[46], Attneave[47], and Brunswik[48] have all stressed that the brain performs statistical computations in one form or the other, and an important use of it

is establishing the statistical regularities of the environment predictively and to adapt behavior to future events[49]. It is important to note that “all learning could be regarded as the internalization of environmental regularities” [49], an important fundamental basis of human knowledge is knowing what is normal and what is abnormal, in other words, humans constantly perform a statistical modeling of the environment and establish a normal environment, and have ways to compare things experienced to the normal environment. It is this strong sense of the customary that capacitates human mind to identify something that is out of the ordinary, sometimes even in subconscious state. And for these reasons, such principles must be effective in detecting cyber intrusions. This perhaps is one of the reasons why statistical features have been widely used in cyber intrusion detection datasets.

Statistical features of network traffic are of vital importance for traffic analysis, traffic categorization and in detection of cyber threats. Statistical features have been widely popular in the research relating to network traffic, and some of the previous research is discussed below.

For years many feature-based traffic classification methods have been using statistical based methods [50].

Taylor in [51] constructed an application called AppScanner for automatic identification and real-time categorization of Android application from within encrypted network traffic, where they used higher order statistical properties of packets sizes in the different network flows, and used this in their algorithm.

Peng in [52] focused on the evaluation of statistical features. In their study they considered six different feature sets including payload data, hybrid feature sets and pure



statistical feature sets, which then was evaluated using three data sets over 10 different machine learning classification approaches. The results were in favor of using statistical features, showing that a strong classification performance can be attained even with a few small statistical features.

Xu et al. [53] in his study developed ways to signify best features to use in statistical-based traffic categorization, and also highlighted on the proper use of higher order moments like skew and kurtosis.

Korczyński et al. [11] used statistical features from headers to classify encrypted traffic. They created statistical features of the information embedded in SSL/TSL headers on traffic of the applications and applied Markov Chain Fingerprinting classification of application traffic and attained satisfactory true positive rate.

De Montigny-Leboeuf [54] in their study described the significance of detecting traffic using traffic flows and using flow features. He elaborated the process of deriving flow features from encrypted traffic, which he later used to identify the application type (interactive typing or data transfer) from encrypted traffic. Moreover, the author provided a list of recognition criteria for several traffic type, including HTTP and HTTPS web browsing traffic, IMAP, POP, SMTP, SSH, Telnet, rlogin, FTP command and data, MSN chat and TCP audio streams. It also provided 39 traffic features that can be used in identifying the traffic types better.

Authors in [55] designed a method called Statistical Protocol Identification to identify the different types of communication (voice call, SkypeOut, video conference,

chat, file upload and file download) within skype traffic, including voip call, conference calls, file download/upload, and normal chatting.

In [56], authors described a Network Intrusion Detection System (NIDS) that detects infiltration or attacks from within encrypted traffic in real time. The NIDS has two parts; the first one clusters the data and forwards to the second engine and the second engine identifies the attack pattern. The authors used the detected anomalies to identify the botnet masters.

Korczynski and Duda [57] proposed using stochastic fingerprints based on Markov chains for identifying application traffic in SSL/TLS sessions. The method uses payload statistics from SSL/TLS headers. 12 representative applications such as Twitter, Skype and Dropbox were used.

### **2.3 Cognitive Cyber Informatics and Complexity Analysis using VFDT**

Traditional machine learning algorithm lack in multiple areas when it comes to detecting modernized malware. This happens for multiple reason explained in [58]. Firstly, the signatures of the various features of the malware changes quite frequently to infiltrate current detection methodologies. The cyber attackers are on constant pursuit to avoid the measures taken by the white hat defenders to secure the system, and for that purpose they tend to change the pattern of the malware to bypass the existing security filters. Secondly, it is difficult to rely on the characteristics of internet traffic data which is so diverse and can come through various open routes and links within the vast internet space, and in various traffic patterns. Hence such algorithms work better on distributions that remain statistically stable. Thirdly, the new generation of attacks specifically target the highly used

protocols and services that cannot be blocked since majority internet traffic utilizes these protocols and services.

There has been a lot of research focus on how humans learn and adapt to problems. The mental learning processes in human brain makes it equipped to perform complex tasks, orchestrate all the resources simultaneously to achieve a goal even without prior first-hand experience, and solve never-seen-before problems based on concepts learned previously. Hence it is implied that imitating the computation mechanism within the brain can help machines do a lot of tasks on their own. The science that focuses on applying the human brain's learning models to other research areas is called cognitive computation and informatics. The main research goal from this process is to upgrade and fine-tune existing critical learning algorithms to make it more adaptive and effective. This process requires the study of several fields including but not limited to computer science, probability, signal processing, artificial intelligence and bioinformatics. It has also been discussed that human beings have an exceptional ability to sense adversities and threats [59], such capabilities are highly effective if applied in designing cybersecurity solutions.

Human beings take input from multiple sensory organs to assess the environment, their brain processes these signals to acquire knowledge about the environment. The extraction of information using their sensory organs can be compared with the process of feature extraction from the data signals. There are infinite number of discriminable different stimuli that a child can get exposed to. A basic function of all organisms is to break the environment into classifications, yet there have been little explicit attempts to particularly understand how a human brain organizes the environment and learns to group objects / stimuli of different forms, shaped and sizes together [60]. The research at University of

California, Berkeley [60] suggests that for each class of objects, there is a basic level for the object which is the most inclusive level, which in other words represent an ‘average’ member of the class, and is the most abstract form in which the object is recognizable. or a normal group by doing complexity analysis of the group. We argue that one of the primary analysis that the human brain does to create this representation is by utilizing the concepts of complexity (discussed in detail in the following sections), which have been defined as an important aspect of human cognitive operations[61]. The concept discussed above is in coherence with our approach, where a group of events are clubbed together, and an estimation of the group in the form of statistical attributes is extracted as features, which helps to create the basic inclusive level for the two classes, one of which is attack and the other is normal, and then the cognition inspired model identifies whether it is an attack group.

Cognition, in a literary sense means knowing, perceiving or conceiving as act. In the computer science language, it can be described as a state of mental process intervening between stimuli and output response, the processes can be described as algorithms that lend themselves to scientific investigations [62]. Cognitive systems tend to be aware of the environment consisting of machines and human beings, where they base their efforts on the knowledge of the physical world and work towards their goals [61]. In systems theory, complexity means a large number of interacting elements with many degrees of freedom, whose individual behavior could not be traced back or predicted [61]. Such systems exhibit self-organization (emergence), thus leading to new features that did not exist before and were not programmed to evolve. The author in [61] have categorized objects, systems and

structures into two distinct types, one containing any order or pattern in their behavior and the other class that obeys no pattern at all and displays stark randomness.

Dimension of an object is the estimation of the convoluted nature of the object. Two very popular types of dimensions include, ED (Euclidean Dimension) and TD (topological dimension). ED represents the minimal number of dimensions in the Euclidean space, required to fully embed the object. And TD is the dimension required to obtain the same object after it undergoes all possible distortions keeping its vital properties intact [63]. Interestingly, both these popular approaches can measure complexity but are limited to integer-dimension space only. However, with the discovery of continuous but nowhere differentiable curves by Weierstrass, and dusts by Cantor and Julia, the notion of the integer dimension had to be refined. These are multiscale fractal objects that exhibit their complexity in terms of non-integer dimensions, which was later popular by the name of fractal dimension. Fractal dimensions of a self-similar or self-affine object is the critical exponent in a power-law relation which makes a measure of the object constant, it is the degree of roughness, irregularity or singularity of the object [64]. Fractal dimensions are calculated by computing the log ratio of minimum number of volume elements with respect to available size to cover an object at multiscale. The log ratio between volume elements and available elements is also known as an exponent which represents the degree of irregularity present in a certain object. Therefore, fractal dimension is proportional to the complexity of a system. In order to find out the morphological, entropy-based or variance-based fractal dimension of an object, crucial exponent of various statistical analysis quantities, and mutual information, entropy, self-similarity, variance is calculated at different scales. It has been established that for anything being of ideal shape or form, i.e.

having smooth surfaces like a line or a circle, the value of the exponent is equal to the integer or topological dimension. Whereas for a rough or irregular object the exponent value is always greater than the TD dimension. Mathematically, the equation for calculating fractal dimension is

$$D_s = \lim_{r_k \rightarrow 0} \frac{\log(N_k)}{\log\left(\frac{1}{r_k}\right)}$$

The fact that there are no fixed feature sets that are suitable for isolation of all type of attacks, makes it evident that we need to explore novel regions of research. Fractal analysis is particularly suitable because, using fractal analysis, continuation of long-term correlation within various scales of the object can be captured. Hence using multifractal analysis, the unique relationship of the pattern in attack landscape can be isolated mathematically. Past research has shown great results when fractals were applied into neural network, better learning was attained in comparison with traditional learning methods [65] [66] [67]. Also the authors in [68] have utilized fractals based k-Nearest Neighbor algorithm to detect latest advanced persistent threats and were successful. Henceforth, fractal-based approaches have proved to be of vital significance in isolating advanced and persistent threats.

## Chapter 3 Proposed Algorithm

The algorithm proposed falls into the category of cognitive based anomaly-based IDSs and assumes that traffic with network intrusion can be differentiated from normal traffic using advanced cognitive machine learning techniques, and that there is no filter-based mechanism that can be used to differentiate attacks from normal data. It focuses on extraction of feature vectors from statistical analysis of the network flows to train the machine learning algorithm for classification. Each feature on the collected data is processed, and their statistical parameters like packet count, mean, median, mode, and other statistical features are extracted to construct a feature dataset. The feature dataset is much smaller in size and captures an estimation of the data flows over time.

### 3.1 Data Cleaning

Our research procedure starts by analyzing the dataset – assessing if there are any irregularities within the raw data, like missing or irregular fields. Each attribute is analyzed and inspected for any missing data. These operations to check irregularities within the raw file needs to be done at the file level, and the commands are run on the Linux operating system on the raw files. This is to stop these exceptions from creating issues within the actual program code. Below is an example of irregularity found within the UNSW dataset. Figure 3 shows how one of the anomalies were dealt with. The character ‘-‘ was found in the raw data and the character’s sequence along with the adjacent characters in the file causes an error in the python script. That is why this anomaly needs to be dealt with in advance before the rest of the data cleaning may take place. This issue is taken care of with

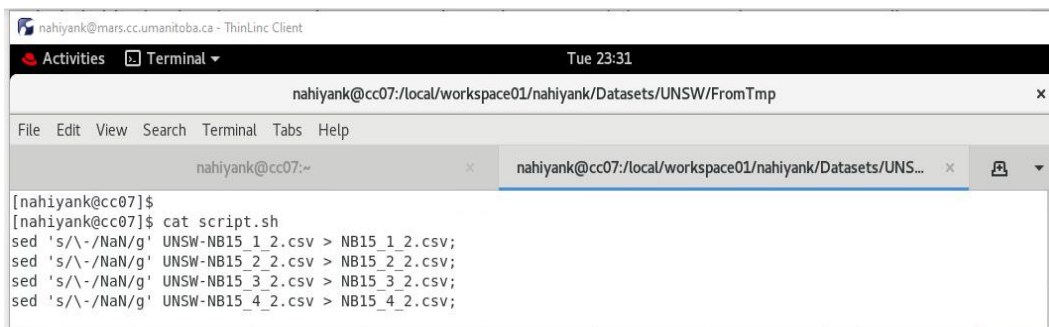
a script that uses sed [3] command as shown in script.sh in Figure 4 to replace ‘-’ characters with another pattern, for example “NaN” and when that pattern is added into the pandas [69] data frame creation statement as a pattern in the na\_values section.

```
df = pd.read_csv('filename', names = [ ], na_values= ["NaN"] )
```



```
nahiyank@mars.cc.umanitoba.ca - ThinLinc Client
Tue 22:38
nahiyank@cc07:/local/workspace01/nahiyank/Datasets/UNSW/FromTmp
File Edit View Search Terminal Tabs Help
nahiyank@cc07:~
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$ grep -Rici "-" .. | awk -v FS=":" -v OFS="\t" '$2>0 { print $2, $1 }' | sort -hr|
grep './UNSW-NB15_1.csv'
467961  ./UNSW-NB15_1.csv
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
[nahiyank@cc07]$
```

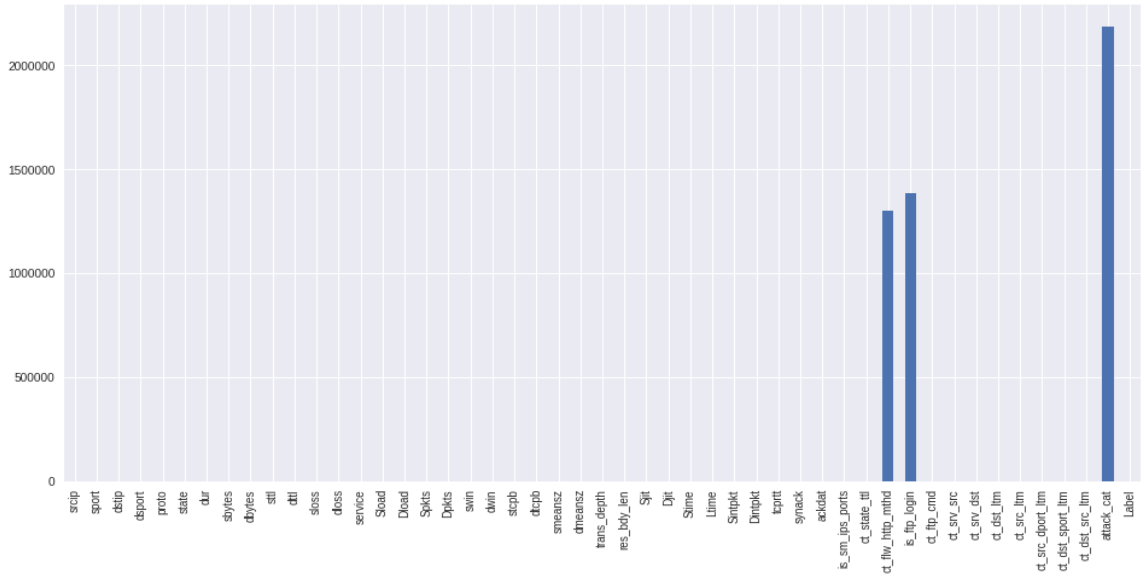
FIGURE 3: DATA CLEANING - GETTING RID OF ANOMALIES



```
nahiyank@mars.cc.umanitoba.ca - ThinLinc Client
Tue 23:31
nahiyank@cc07:/local/workspace01/nahiyank/Datasets/UNSW/FromTmp
File Edit View Search Terminal Tabs Help
nahiyank@cc07:~
[nahiyank@cc07]$
[nahiyank@cc07]$ cat script.sh
sed 's/\-/NaN/g' UNSW-NB15_1_2.csv > NB15_1_2.csv;
sed 's/\-/NaN/g' UNSW-NB15_2_2.csv > NB15_2_2.csv;
sed 's/\-/NaN/g' UNSW-NB15_3_2.csv > NB15_3_2.csv;
sed 's/\-/NaN/g' UNSW-NB15_4_2.csv > NB15_4_2.csv;
```

FIGURE 4: SCRIPT TO REPLACE IRREGULARITIES FROM THE FILE





**FIGURE 5: MISSING DATA - UNSW**

### 3.2 Missing Data

Missing data represents the fields within the dataset that has no values in some of the rows in the dataset. These represent cases where the attribute's values were captured incorrectly or simply not captured at all. Such cases of missing data are common in datasets and dealing with them is a must. Below are the missing data statistics for the two datasets, AWID and UNSW. Figure 5 shows that the columns `ct_flw_http_mthd`, and `is_ftp_login` has a null value in most of the rows in the dataset. In the later steps this field will be removed entirely as this attribute doesn't contribute much anyway.

These null values in attributes `ct_flw_http_mthd` and `is_ftp_login` are imputed with -1, which is a common practice when using the scikit learn [70] libraries. The same procedure is performed for the both datasets, AWID and UNSW.

The AWID dataset captures which contains wireless data, which unlike wired connection dataset, contains several flags for wireless communication, many of these attributes were not possible to capture in this dataset. Below are the missing data stats for the AWID dataset.

As we can see from the figures, there are major number of attributes for this dataset with missing data. This significant portion of missing data will need to be dealt with before the dataset before a machine learning algorithm can be used on it. In order to advance to the next stages, we can simply impute the values. We are selecting the median imputing strategy. This imputing strategy is of little significance because our machine learning algorithm will be working on the extracted features, which are the statistical properties of data streams, and not the actual values in each row. The imputation library in python / pandas is used for this purpose.

### **3.3 Data Encoding**

After imputing the data and getting rid of the missing values, and other irregularities within the dataset, the numerical attributes in the dataset can undergo the statistical computations and machine learning operations. However, the attributes that are non-numerical will need to be converted to numbers, i.e. the non-numerical or categorical values will need to be replaced with numbers so that the mathematical operations and the machine learning models can be executed on them. If these values are not dealt with, and only the numerical values are worked upon, the machine learning model will lose significant information that could have been useful to differentiate the attack data from the

normal. Scikit-learn's [70] label encoder library function is used for this purpose. A fit-transformation operation is performed on each of the attributes containing non-numeric data. The list of categorical attributes are as follows:

**TABLE 1: LIST OF CATEGORICAL ATTRIBUTES- UNSW**

srcip
dstip
proto
state
service
attack_cat

### **3.4 Feature Engineering**

In this research, I have incorporated some feature engineering techniques to improve the detectability of the dataset, these approaches, along with the rationale behind, have been elaborated in this section.

The dataset for UNSW contains 49 columns in total. One of the aspects in the dataset that needed to be improved was that the representation of the host addresses. In a standard network, the addresses are identified using the host address and the network address, in other words, the subnet that the host belongs to. There was no subnet information for the involved IPs, and if this information was not added into the dataset a critical domain-specific information would have been missing. After adding the subnet, two hosts will have same subnet mask even though their specific IPs are different, which will provide

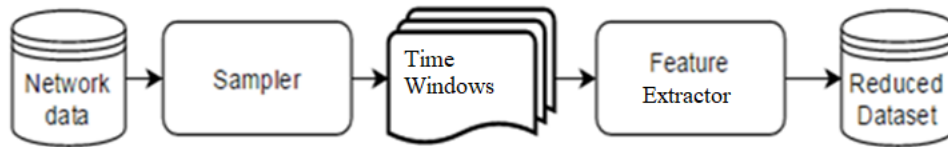
additional information for the machine learning algorithm to relate these two IPs. For this purpose, two new fields were added - `srcip_trunc_encoded` and `dstip_trunc_encoded`, which only contains the network portion of the address and ignores the host portion if the IPv4 address.

Another aspect of the dataset that needed to be fine-tuned was how the addresses were encoded before feeding to the machine learning algorithm. As you would have assumed, when we encode source IP like any other attribute, we create a dictionary from all the values that are. Since these two fields/attributes are interconnected, we encode these IPs to number values from a single dictionary created for all the IP. Similarly, there are other fields like `'wlan_ra'`, `'wlan_da'`, `'wlan_sa'` and `'wlan_ta'`, which represents some network addresses. It seems contradictory if the same address is represented by a certain value in the source address field and represented by a different value when it appears in destination address field. That is the reason why these fields are created from the same dictionary which contains the entire list of all the addresses that appear in the network trace.

### **3.5 Sampling of the Network Dataset**

When working with large datasets, especially in the field of analyzing network traffic, human analysts often divide the dataset into smaller fractions that can be analyzed individually in further detail. Separate synopsis can be extracted from these small fractions that can be used to construct a depiction of what has happened within the time fractions. For our algorithm to be fast and responsive, we would need to have a much-reduced data set. Hence, we club several network events into a single representation, based on their

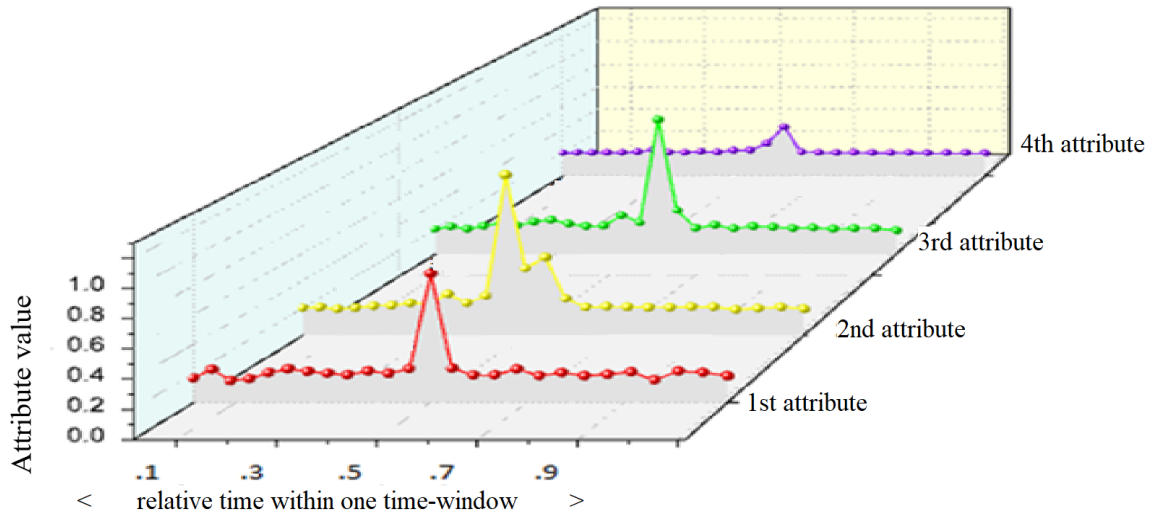
adjacency in time, i.e. if the events occur consecutively in time, we club these events into a single representation. In our proposed algorithm, this job is done by the component called sampler. Our sampler divides the large dataset into time-windows, fragmenting the data set into smaller sections based on time. In terms of dataset rows, after the clubbing, each row of the reduced dataset will contain information from several data rows from the original dataset, which is elaborated further into this section. The approach is depicted in Figure 7.



**FIGURE 6: SAMPLER AND FEATURE EXTRACTOR**

Figure 7 shows how an attribute is divided into time-windows, with an example. Let us imagine a dataset having 5 attributes, one of the them is the timestamp, which means the time when the values of the 4 attributes were capture, much like our dataset. In the below figure we plot these 4 attributes against time. The leftmost point represents the first timestamp within this time-window and the rightmost point represent the last captured value of the attribute within the time-window. Notice that there are 30+ points in the red time-window. This means the 30 rows had a value of  $S_{time}$  which corresponds to the first time-window and hence, the entire time-window these 30 rows will be represented by a single time-window, and after we extract higher order statistical features for the entire time-

window, that will represent the entire 30+ points. In the next section we will see how these time-windows will be processed and features from them will be extracted to create the reduced feature dataset.



**FIGURE 7: TIME-WINDOWS ELABORATED**

An important topic to discuss here is the concept of stationarity. A signal is considered stationary if its statistical moments do not change over time. Some of the conditions for the signal to be stationary is the mean and variance should be constant over time, and there should be no seasonality or repetition of signals. If they don't change at all, the signal is considered strongly stationary, if it changes slightly, it is called weakly stationary. If the signal is strongly stationary, the statistical moments of the entire signal can be obtained from any portion of the signal. Often, working with time series, the concept of stationarity is important, specifically when the time-series is analyzed for predictive purposes.

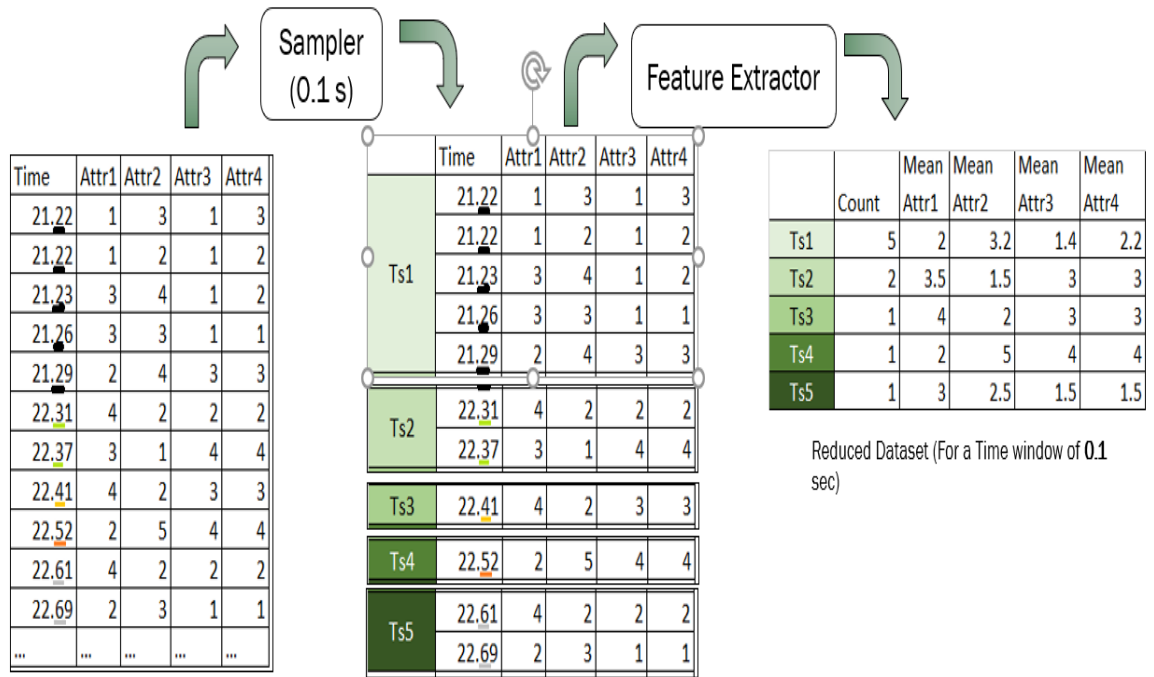
Stationarity is an important factor when it comes to statistical inferencing and modelling, it tells the audience whether some of the models like AM model and the MA (Autoregressive–moving-average model) can be used or not.

However, in this part of our approach, we have used descriptive statistics, which is a process of using absolute numbers to summarize the data samples. Descriptive statistics are most helpful when the research is focused towards expressing the data population in summarized forms.

### **3.6 Feature Extraction**

These small segments or time-windows are then processed, and higher-order feature vectors are constructed from them. Each of the time steps consists a few rows of the original dataset. These small segments or network data flows are now processed to extract features from the statistical analysis of the group. The purpose is to extract key information from this group of events and decrease the amount of data from the initial dataset. Our hypothesis is that despite the decreased amount of data, the information represented will be powerful enough to separate attacks from normal using machine learning approaches. Now, the feature extraction portion of the algorithm extracts the below statistical properties for each of the attributes within the group. It is meaningful to say that these statistical attributes capture all the details from the sample. The process of the feature extraction is elaborated in the below figure. The first table contains the sample rows from the original dataset. The sampler divides the dataset into smaller time-windows, each time-window consisting of traffic from  $1/10^{\text{th}}$  of a second. All the rows with the ‘Stime’ value corresponding to 21.2

will go into the first time-window and every row corresponding to 22.3 will go into the second time-window, so and so forth.



**FIGURE 8: FEATURE EXTRACTION ELABORATED**

Now, the feature extractor extracts the below mentioned statistical higher order features from the time-windows presented. The rightmost table in the above figure corresponds to the mean feature extracted for each of the attributes in the dataset, in the same manner all the other features like median, mode, variance, maximum, minimum, standard deviation and other statistical values will be calculated for each of the attribute. And the final addition will be the count of rows in each time-window, which will be one per time-window. Hence the reduced dataset has one row per time-window, which decreases the number of rows



significantly, but at the same time the number of columns increases 8 times plus 1 for the count.

We have tried to include as many features as possible to extract as much information as possible. We believe that these higher order features are enough to successfully capture all the significant events happening with each attribute within this period. Below are the features explained.

I) Mean

Mean,  $\bar{X}$ , is the summation of all the observation values divided by the number of observations

$$\bar{X} = \frac{\sum X}{N}$$

where  $X$  = an element of the sample

$\bar{X}$  = the mean of the sample

$N$  = the number of samples

II) Median

Median,  $\tilde{X}$ , is the middle value in a set. If the number of the observations is odd

Median is the value at position :  $\left(\frac{n+1}{2}\right)$

If the number of observations is even, the Median is the average between of the

values at positions :  $\left(\frac{n}{2}\right)$  and  $\left(\frac{n+1}{2}\right)$

Where  $n$  = the number of samples

III) Mode

Mode is the value that has repeated the most within the sample. If there is no repetition, there is no mode, and we make the value 0.

IV) Variance

Variance is a measure of how further spread out the population is around its mean.

$$\sigma^2 = \frac{\sum(\mathbf{X} - \bar{\mathbf{X}})^2}{\mathbf{N}}$$

$\mathbf{X}$  = the sample value

$\bar{\mathbf{X}}$  = the mean of the sample

$\mathbf{N}$  = the number of samples

$\sigma^2$  = variance

V) Maximum

The maximum value from the sample

VI) Minimum

The maximum value from the sample

VII) Standard Deviation

Below is the formula for standard deviation of the sample:-

$$\mathbf{S} = \sqrt{\frac{\sum(\mathbf{X} - \bar{\mathbf{X}})^2}{\mathbf{N}}}$$

where  $\mathbf{S}$  = the standard deviation

$\mathbf{X}$  = value of an attribute in each of the row of the dataset

$\bar{X}$  = mean of all the values in the dataset

#### VIII) Count

Count is the total number of packets within the specific time-window.

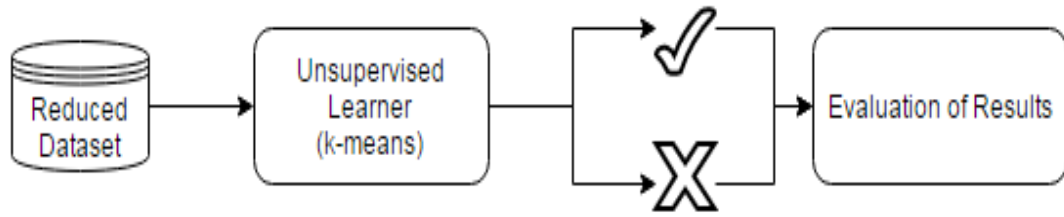
Although the extracted dataset has more columns which expands the dataset horizontally, however, the dataset is reduced significantly on the vertical axis. Each row now replaces several rows representing concurrent events on that time window. And these statistical features are important to identify the pattern of the flow of each of the attribute in this time-window, and without this critical information, it will be difficult to understand and identify the changing behavior of each of the attribute within this time-window. Once we have all the data streams and the statistical snapshots of each of the attribute is collected, we will have a smaller dataset with each of the events captured with enough detail. The statistical analysis of each of these attributes are extracted as feature and a new data frame is constructed from these feature vectors, this dataset will be used to further identify the time-windows that contains attack events. With a smaller dataset to operate upon, the final machine learning algorithm will require fewer computations and will require less time to converge and produce results. Since we have put all the statistical details of the flow of each of the attributes in the new dataset, it is expected that the loss of information is much less, despite the significant reduction of the dataset.

After extracting the statistical feature vectors and constructing with them the new reduced feature data set, let's focus on what type of machine learning algorithm we will be applying to separate events into two distinct groups. By now we have already made our algorithm less expensive in terms of computational power, the data set that we must

work upon is much reduced in size. It is noteworthy that high accuracy from unsupervised detection is rare, and a Cybersecurity attack detection based on unsupervised learning can't solely depend on such a model that is incapable of delivering best results every time. Hence our model is not focused towards accuracy and isolation of the attack element, rather it is more geared towards a fast response, simplistic implementation and towards allowing the human analyst to avoid time consuming inspection of normal traffic and help them focus on actual problematic attack traffic. Our model only narrows it down to a time frame which seems anomalous and further human analysis is required by the incident response team to isolate which event in the time frame is anomalous and needs to be resolved eventually.

### **3.7 Unsupervised Classification Using K-means Clustering**

Our Unsupervised algorithms need to make inferences without any training or reference of the known or labelled examples. K-Means Clustering algorithm is a good choice for the unsupervised model, it is an np-hard iterative refinement technique that aims to perform k partitions within the n samples presented. It is surprisingly versatile and hence it is frequently applied in many fields including computer vision, protein enhancement, vector quantization and computational geometry.



**FIGURE 9: SAMPLER AND FEATURE EXTRACTOR [71]**

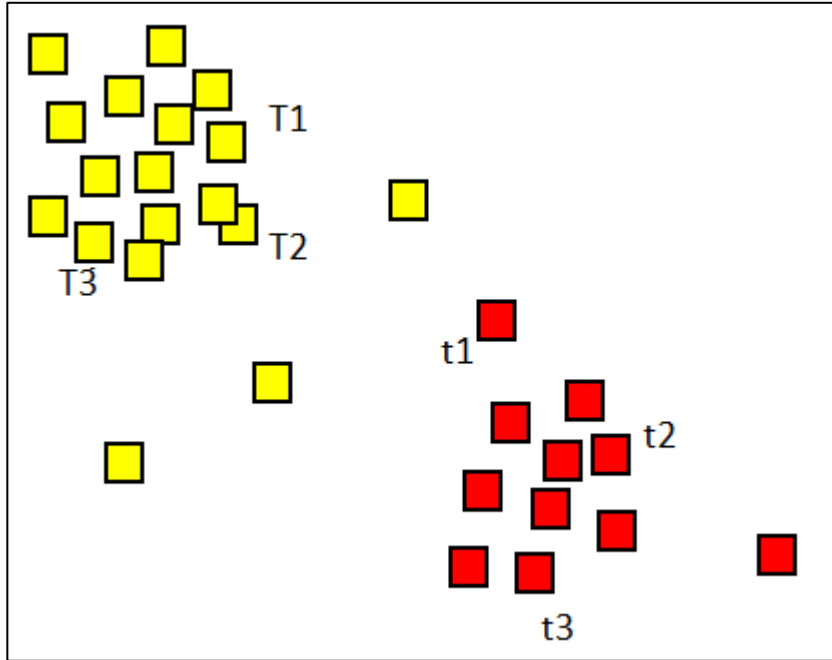
The algorithm tries to organize the feature space into ‘k’ number of clusters, and intends to minimize the distance between points of each cluster.

### **3.8 Attack Detection**

We have isolated the traffic into two clusters, and we are claiming that one of the clusters represent attack traffic and the other cluster is comprising of normal traffic which can be discarded from intrusion analysis. However, at this point we are unable to concluded which one of these two clusters is the attack cluster.

We now focus on extending the work further and exploring if there are any further approaches that can accurately identify the cluster that represents attack. As we have explored in the background studies section, there are some advancements done in the research community where researchers have proven that there is a difference in fractal dimension trajectory for network traffic having malicious attacks. We will be using this approach to conclude which of these clusters contain attack traffic so that the human analyst or the incident response team can focus specifically on that cluster and analyze on those timestamps and address the intrusion promptly. We are going to perform complexity

analysis by calculating the variance fractal dimension trajectory (VFDT) for the time-windows. The process is further elaborated below.



**FIGURE 10: A 2D EXAMPLE OF TWO SEPARATE CLUSTER GROUPS**

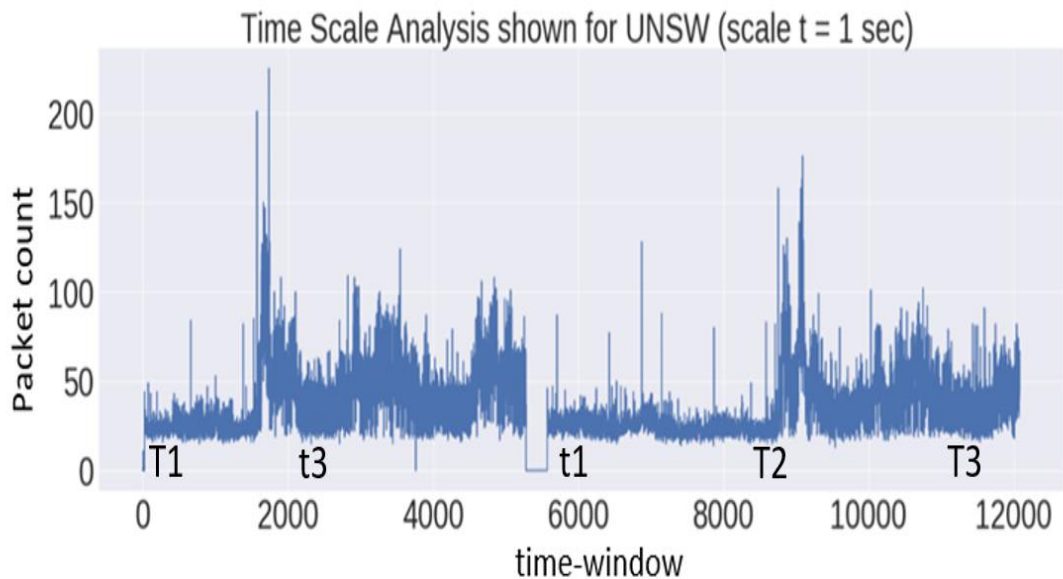
Our data is of several dimensions and cannot be displayed, however, let's take an example for better understanding. In Figure 10, there is a 2-dimensional picture of items separated into two distinct clusters. Let's assume that these are the two clusters that the algorithm has isolated and one of the clusters is the time-windows representing normal traffic and the other is the cluster of time-windows with attack traffic. We need to understand that each of these items represents nothing but a specific time-window, i.e. they each represent a specific range in time. The items in the attack cluster represent a time-window that contains potential attacks and the normal cluster represent time-windows that contains nothing but normal traffic.

As per theory elaborated in previous sections, it has been predicted that attack traffic should have higher complexity, in other words, higher VFDT values. Hence the group that contains greater number of elements containing higher VFDT values should be nominated as the attack cluster. Hence, we must find out the group or the cluster that has more elements with the higher value of VFDT in it. One way of doing it would be to accumulate or add the VFDTs of the elements of each group and compare with each other, and, to nominate the group with the higher sum value as the attack group.

However, instead of going through each individual item in the group, which will be significantly taxing in terms of time and computation, we can perform the same comparison over 10 randomly chosen items from each group. So, from each of the clusters the algorithm should pick an item randomly and compute the VFDT from the original dataset and add it up into the counter where it stores the accumulation sum of the VFDTs for that group. In the end the group with the higher accumulation sum counter value should be nominated as the attack group by the algorithm.

Let the 10 items randomly chosen from yellow cluster include, T1, T2, T3 and 7 other items, and likewise 10 items randomly chosen from the red cluster include t1, t2 t3, for example. After choosing the item, say t1 for red cluster from Figure 10, it retrieves the time that this item corresponds to, and it goes back to the original data set ( refer to Figure 11 below) and computes the VFDT of the traffic data at timestamp t1 from the plot of packet count vs time. Let us consider that the VFDT value for t1 is v1, for t2 is v2 and for t3 it is v3. And similarly, the VFDT values T1, T2, T3, is V1, V2, V3 as depicted in the figure. For t1, the algorithm will add the VFDT value v1 to a counter v, and on the next iteration for the same cluster say for t2, the algorithm will add the VFDT value v2 to the counter v,

and so forth. After ten iterations the counter will contain the sum of the VFDT values. Similarly, the sum of VFDTs will be accumulated in another counter, say  $V$ , and at the end by comparing the two counters  $v$  and  $V$ , the algorithm can nominate the group that has the higher sum as the attack group.

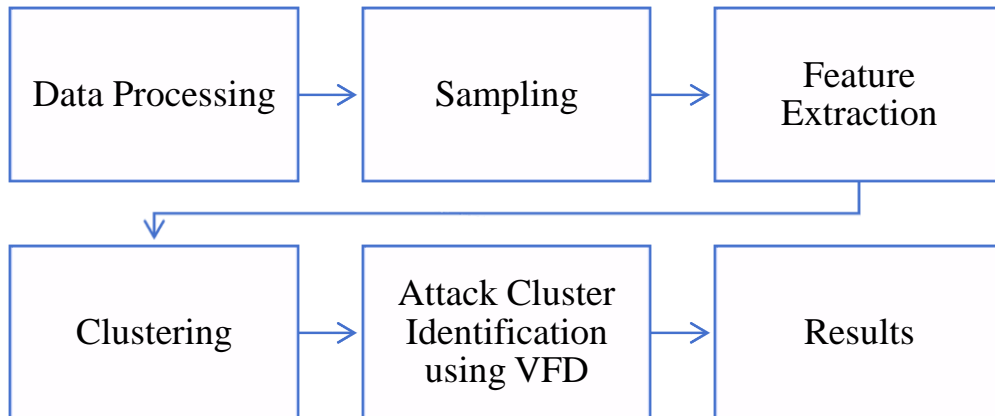


**FIGURE 11: ATTACK ISOLATION EXPLAINED**

### 3.9 Proposed Process Model

Figure 12 shows the final model with the order of processes occurring in this operation from start to finish. The process starts with processing the dataset, cleaning the data from any impurities, imputing, and encoding the dataset into the AI friendly format, and then sampling with the sampler, extracting higher order features, and comes the unsupervised clustering, and afterwards complexity analysis using VFDT analysis that yields the results.





**FIGURE 12: PROPOSED MODEL**

### **3.10 Proposed Network Topological Model**

Now that we have the process model presented, let us explain how these processes can be placed within a live network, and what process fit into which components of an ethernet network. Let us consider a sample network having three hosts and are transmitting packets mentioned in Table 2 within a time-window  $ts_1$ . Within this time-window, the IP1 and IP2 is transmitting 2 packets and IP3 is transmitting one packet.

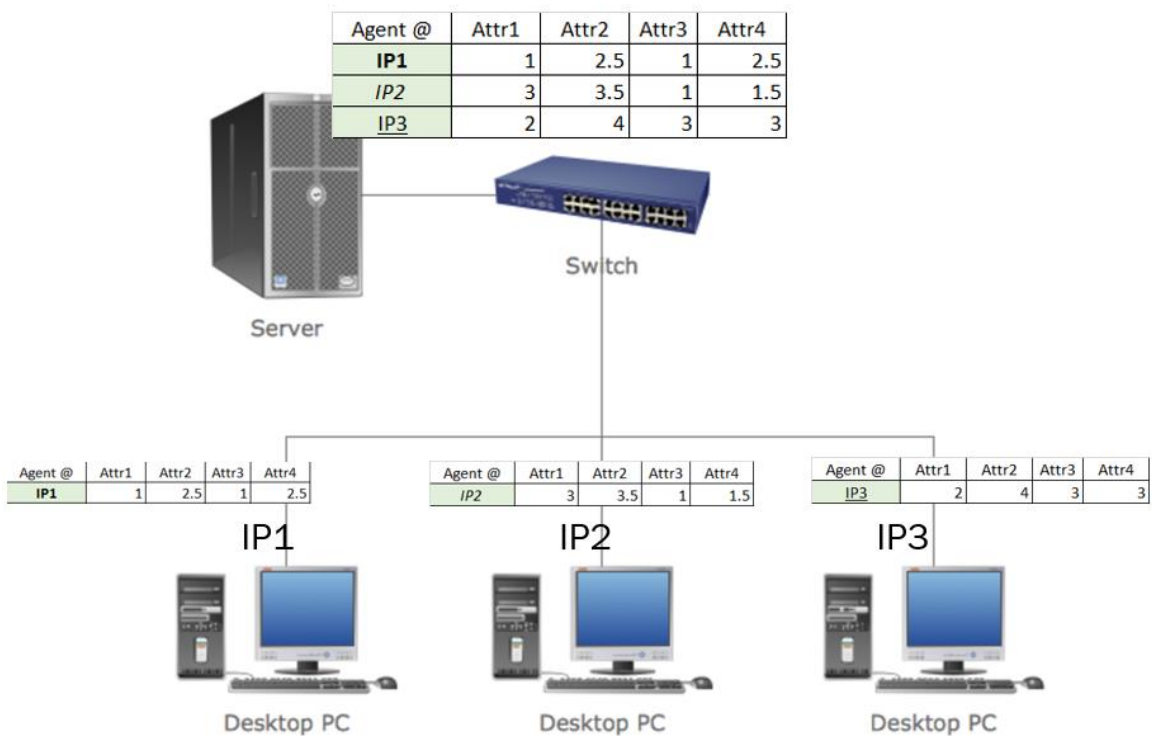
**TABLE 2: NETWORK COMMUNICATION**

	IP	Time	Attr1	Attr2	Attr3	Attr4
Ts1	IP1	21.22	1	3	1	3
	IP1	21.22	1	2	1	2
	IP2	21.23	3	4	1	2
	IP2	21.26	3	3	1	1
	IP3	21.29	2	4	3	3

As per our proposed algorithm, higher order features of each of the attributes need to be extracted. This operation can be done centrally on a server, that we are calling network intrusion detection system (NIDS), will receive all the network traffic from the switch or gateway and calculate these features to perform the rest of the operations. The advantage of incorporating the change in this manner is that all the implementation will occur in a single node. On the other hand, this means that the NIDS server needs to be resourceful enough to analyze the traffic of each second fast enough to cater for the traffic coming in the next second. This also increases the traffic within the network, as literally all the packets flowing into the network will need to be relayed again to the NIDS.

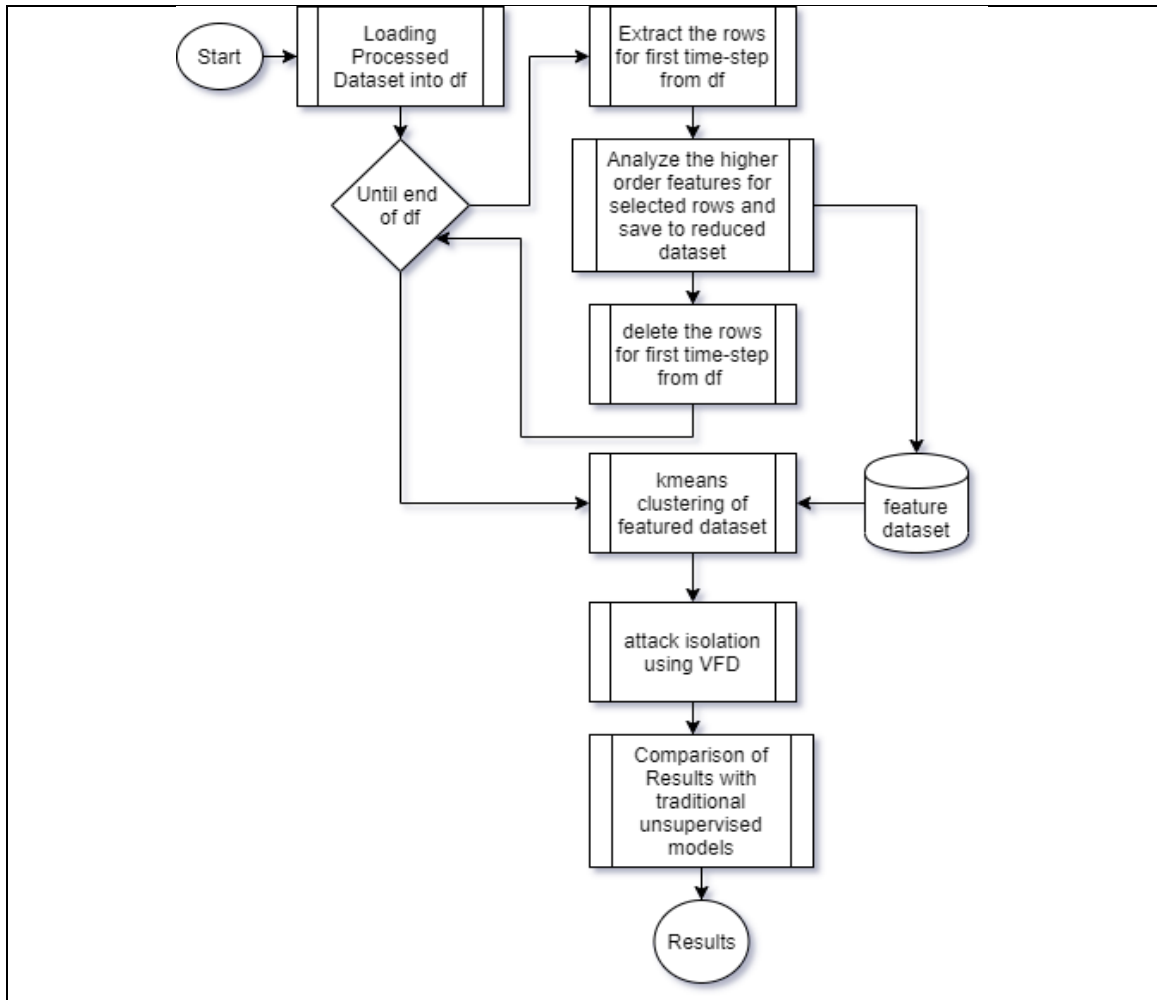
In order to avoid this issue, we can also perform some of the operations on the host itself, and the rest of the operations can be done on the central network intrusion detection system. If some of the operations are done on the host, the computational resource of the host gets utilized releasing some of the load from the NIDS. In order to do that, all the network events (i.e. rows in the dataset) that happens in a time-window is processed by the host itself. And the host accumulates all the traffic rows generated for that time step and extracts the higher order features, and further send to the central network intrusion

detection system just a single row for each processed time steps. This way, the data that needs to be transferred also get optimized and it is better for resource utilization as well. But this means that changes need to happen within each connected host within the network. Which means that there needs to be added a small software into each host that needs to be report to the central host. Figure 13 depicts the hybrid network intrusion detection system reporting the mean of the attributes, where the machines IP1 and IP2 have processed multiple events (refer to Table 2) and passed only a single data packet to the central server.



**FIGURE 13: NETWORK TOPOLOGY EXPLAINED**

### 3.11 Workflow of Proposed Algorithm



**FIGURE 14: FLOWCHART OF PROPOSED METHOD**

In the above flowchart, the algorithm starts with extracting the data rows from the dataset files, pre-processing the data and loading it into a pandas dataframe df. Now it fetches the first time-window's data from df and extract the higher order features for these number of rows. These higher order features are stored in the feature data set. After processing the rows for that time-window and extracting the features, the rows are deleted from df. The

same process repeats where the first time-window's rows are extracted processed and deleted until there are no new data rows to be processed. Once this process is finished, the kmeans clustering is used to cluster the dataset into two distinct cluster. The next process is to identify the attack cluster using VFD calculations, from the original dataset. Finally, the results are calculated further and compared with the traditional unsupervised model's results.

## **Chapter 4 Experiments and Results**

This chapter is focused on the execution of the above-discussed algorithm, later it also elaborates on the results of the experimentation. The chapter is divided in several sections. We discuss the results of each of the portions on both the datasets UNSW and AWID.

### **4.1 System Setup**

This simulation was performed on the University of Manitoba managed IT cluster infrastructure, using one of the 10 clustered Dell PowerEdge R815 server systems, each server contained 4 AMD Opteron 6272 CPUs, each of which had 8 cores and each core served 2 threads. The server was equipped with 256 GB of RAM.

### **4.2 Performance Evaluation**

An important consideration for detection centric algorithm is how to evaluate the results and assess the performance of the detection. An efficient intrusion detection system should be able to capture maximum amount of intrusion attempts with the least number of normal traffic characterized as attack. However, for a completely unsupervised model, as we have emphasized before, attaining such results are very ambitious, especially in the field of cybersecurity, where getting significant accuracy is difficult to attain even with hours of training. For our study we are going to assess the model performance using precision, recall and F1 score. The metrics are explained in the below equations. Here, TP is true positive, which means that the item assessed by the algorithm as an attack is indeed an attack, TN

(True Negative) is when an item assessed as normal is indeed a normal, FP (False Positive) is when the item is actually normal but assessed to be an attack by the algorithm, and a FN (False Negative) is when an actual attack is classified as a normal by the algorithm.

$$\mathbf{True\ Positive\ Rate} = \frac{TP}{TP+FN} \quad \text{EQUATION 1}$$

$$\mathbf{True\ Negative\ Rate} = \frac{FP}{FP+TN} \quad \text{EQUATION 2}$$

$$\mathbf{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} \quad \text{EQUATION 3}$$

$$\mathbf{Presicion} = \frac{TP}{TP+FP} \quad \text{EQUATION 4}$$

$$\mathbf{Recall} = \frac{TP}{TP+FN} \quad \text{EQUATION 5}$$

$$\mathbf{F1} = \frac{2 \times TP}{2 \times (TP+FN)} \quad \text{EQUATION 6}$$

### 4.3 Experimental Dataset

We have considered two data sets to simulate our experiments. We perform the evaluations on UNSW NB-15 and AWID dataset. We consider all the files for UNSW dataset. Our simulation considers everything in the dataset files, all the rows and all the features in the raw files.

For the dataset UNSW-NB15, containing 3,239,993 rows has about 14.48% attack rows and 85.52% normal rows. Figure 15 shows the number of attacks and their types. We can see that there are 10 types of attacks and they are denoted by the attack\_cat field in the dataset.

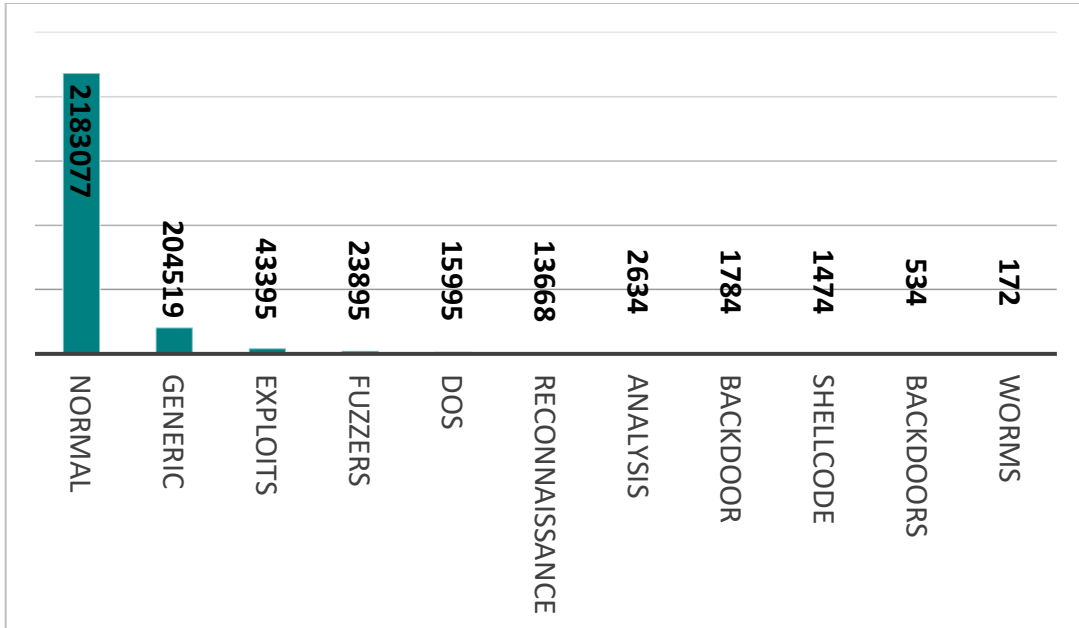


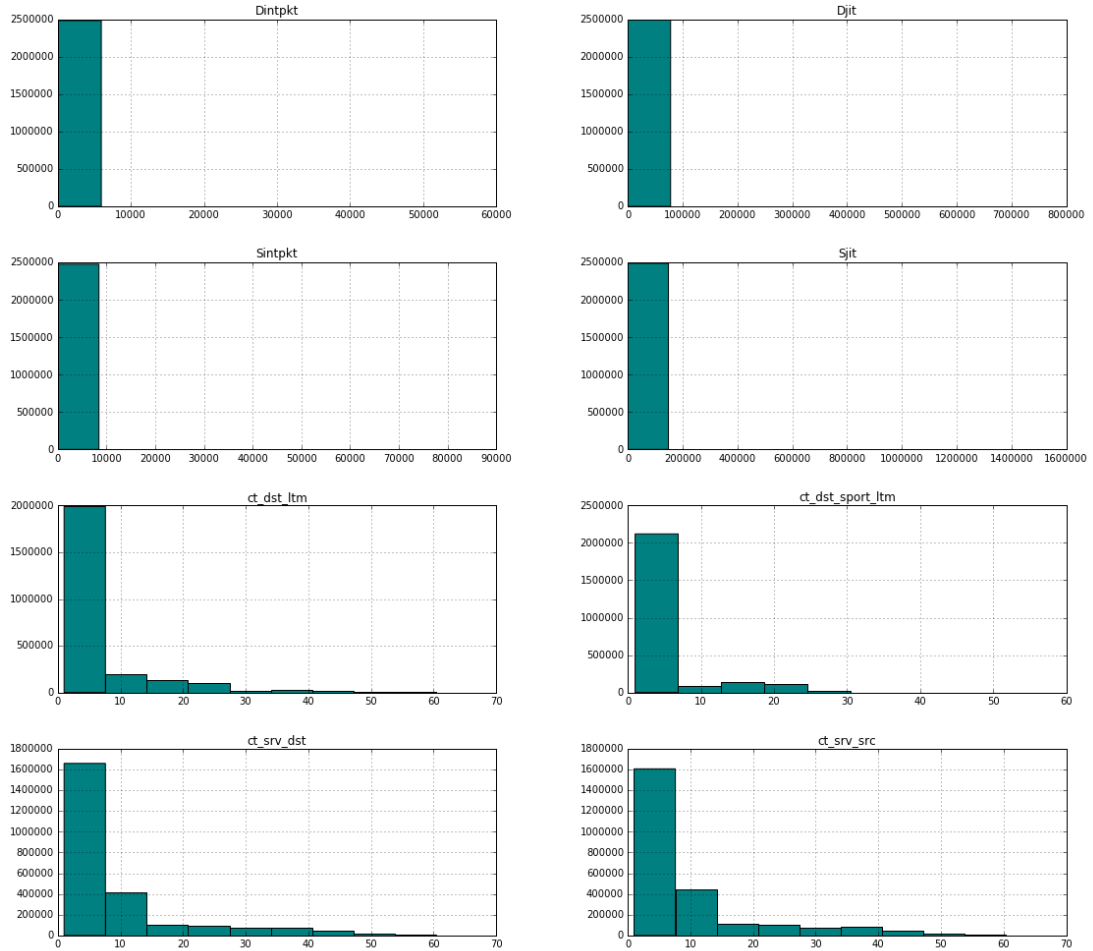
FIGURE 15: ATTACK AND NORMAL COUNTS FOR UNSW DATASET

#### 4.4 Experiment 1: Exploring the dataset

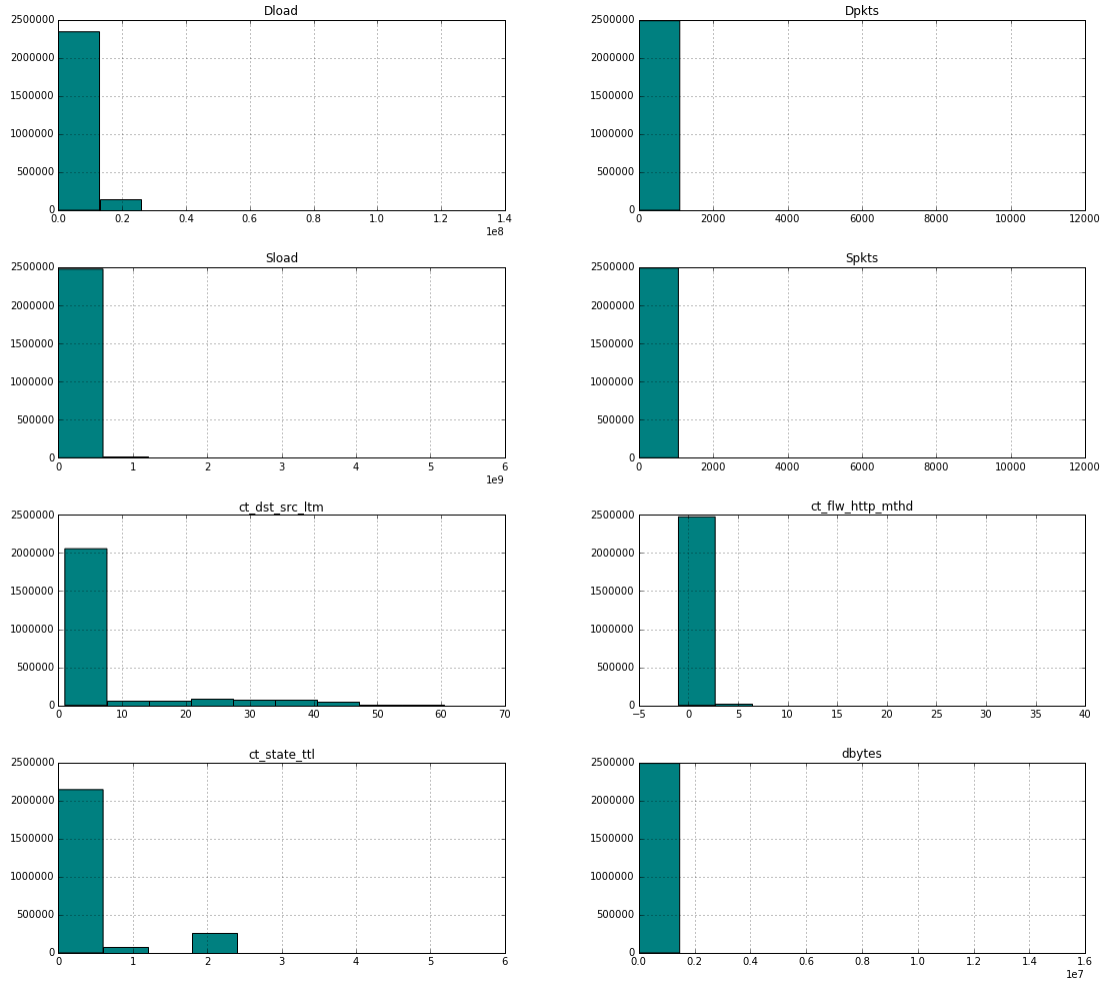
In this experiment we explore the dataset and to understand the various features of the dataset. The figures below will be providing us a high-level representation of the various features. There are in total 47 features in the dataset, most of them are integer attributes, some are categorical attributes, and some contain float values. For the integer and float attributes that have a lot of unique values, we perform histogram analysis, and understand



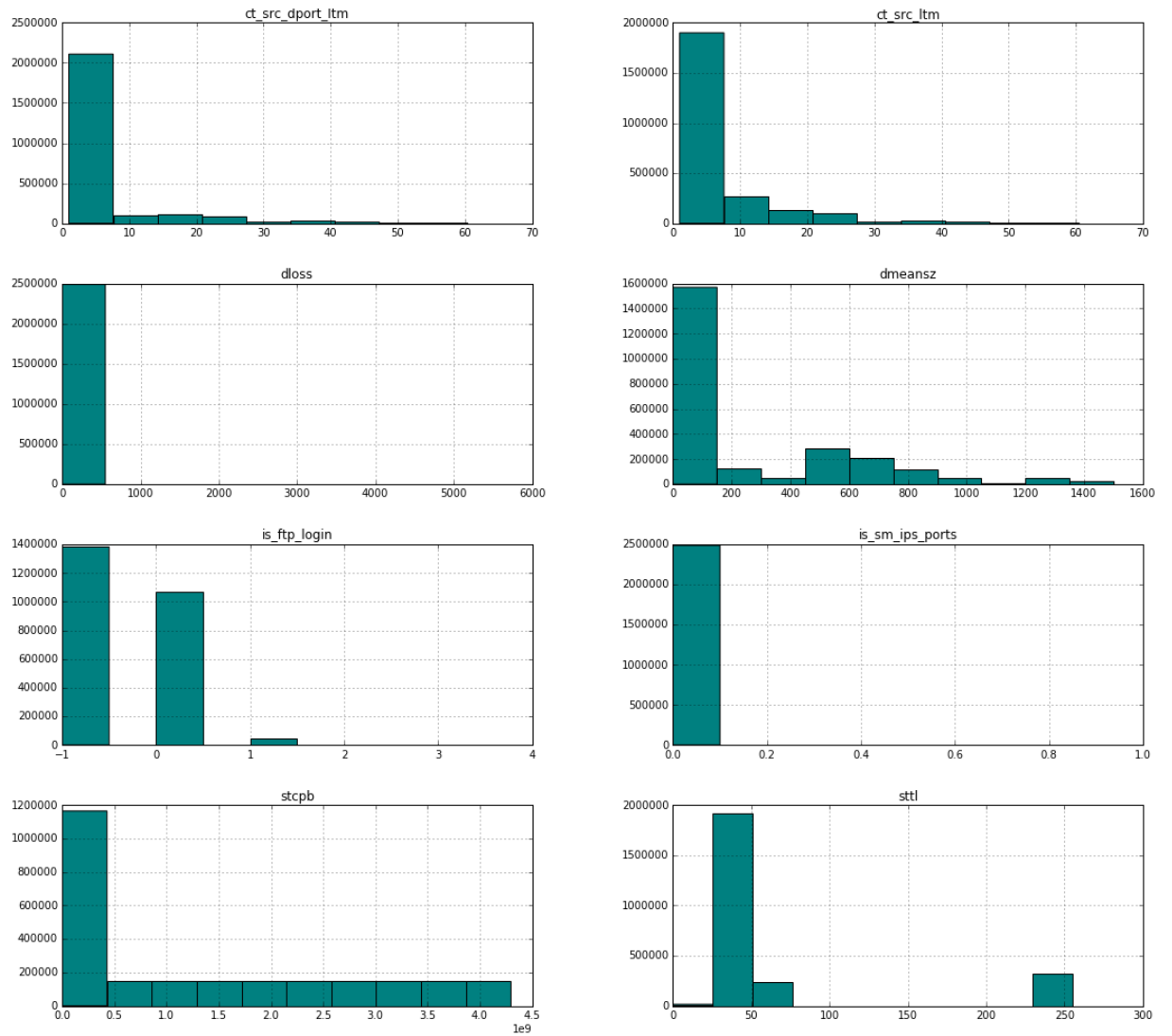
what the range of the values are. For the attributes with a small number of unique values, we can drill down further and perform a value count analysis, which shows us the specific values taken and how many times they appear in the dataset.



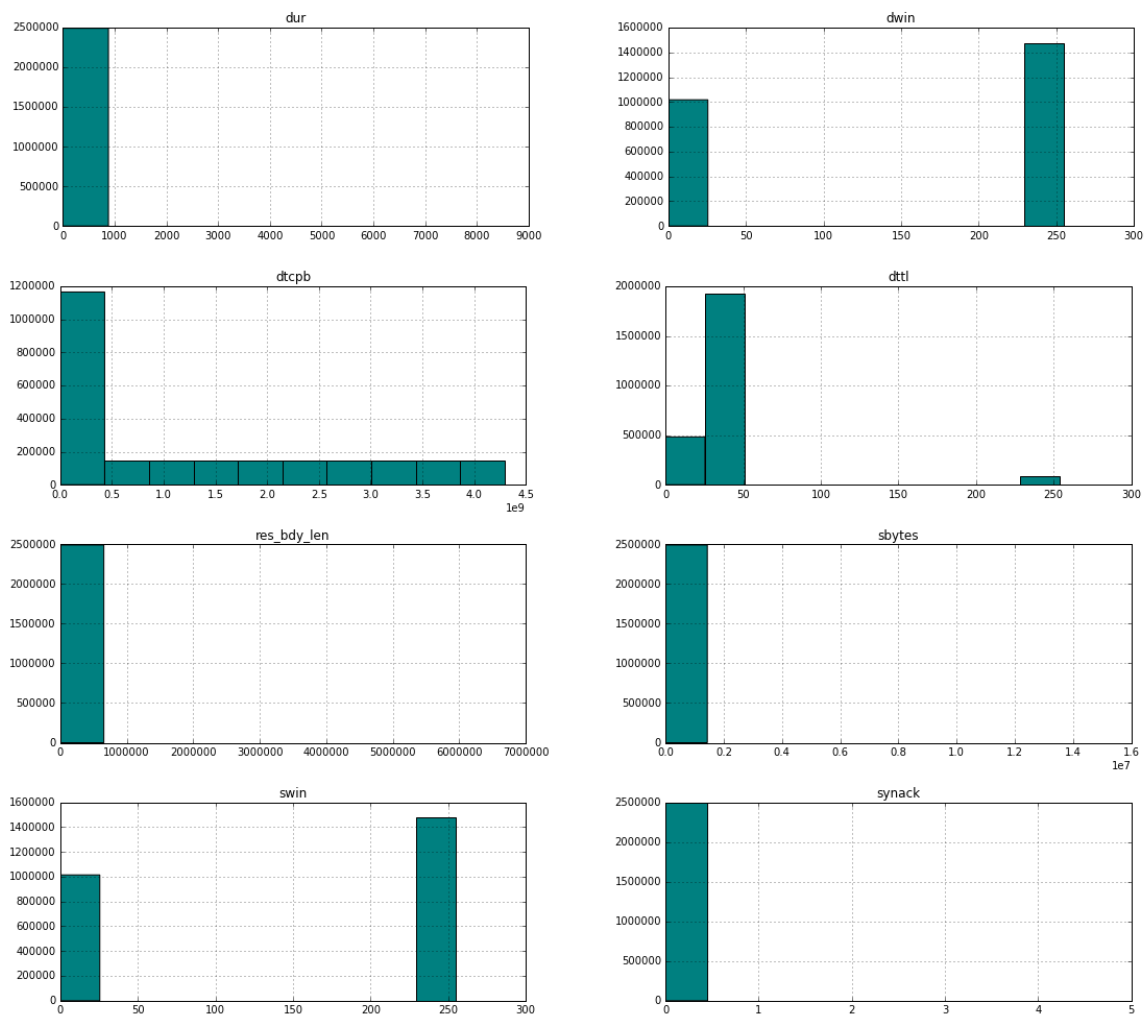
**FIGURE 16: HISTOGRAM ANALYSIS OF FIELDS 'DINTPKT', 'DJIT', 'SINTPKT', 'SJIT', 'CT\_DST\_LTM', 'CT\_DST\_SPORT\_LTM', 'CT\_SRV\_DST', 'CT\_SRV\_SRC'**



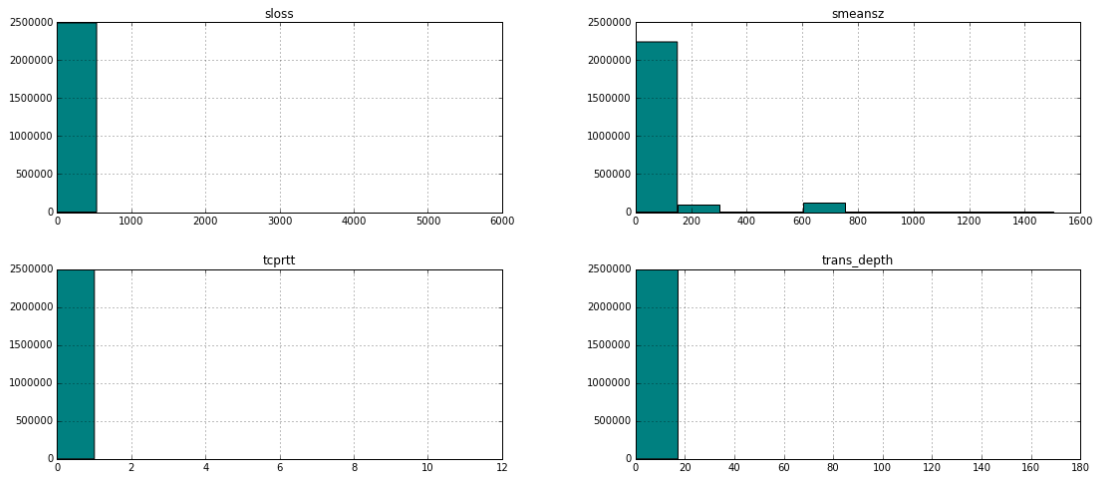
**FIGURE 17: HISTOGRAM ANALYSIS OF FIELDS DLOAD, SLOAD, SPKTS, CT\_DST\_SRC\_LTM, CT\_FLW\_HTTP\_MTHD, CT\_STATE\_TTL, DBYTES**



**FIGURE 18: HISTOGRAMS FOR FEATURES CT\_SRC\_DPORT\_LTM, CT\_SRC\_LTM, DLOSS, DMEANZ, IS\_FTP\_LOGIN, IS\_SM\_IPS\_PORTS, STCPB, STTL**



**FIGURE 19: HISTOGRAMS FOR FEATURES DUR, DWIN, DTCP, DTTL, RES\_BDY\_LEN, SBYTES, SWIN, SYNACK**



**FIGURE 20: HISTOGRAMS FOR FEATURES SLOSS, SMEANSZ, TCPRTT, TRANS\_DEPTH**

Here, let us focus on the features that are represented in the dataset by only a handful of unique values, which keep repeating throughout the dataset. For these features, analyzing the value count charts makes sense because there are only a few unique values. With the value count analysis we will be able to understand the variation of the data within the dataset. For some attributes, like srcip (Source IP) in Figure 24, we will be able to know how many unique hosts are there, and who are communicating to what extent.

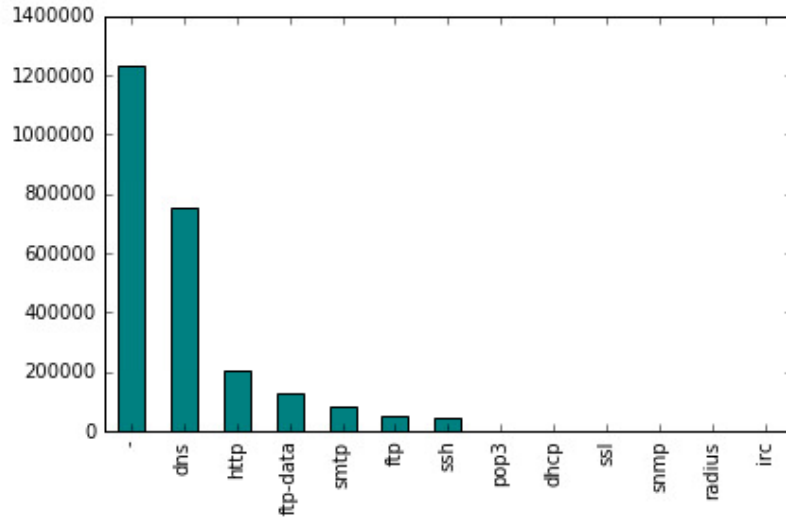


FIGURE 21: VALUE COUNT CHART - SERVICE

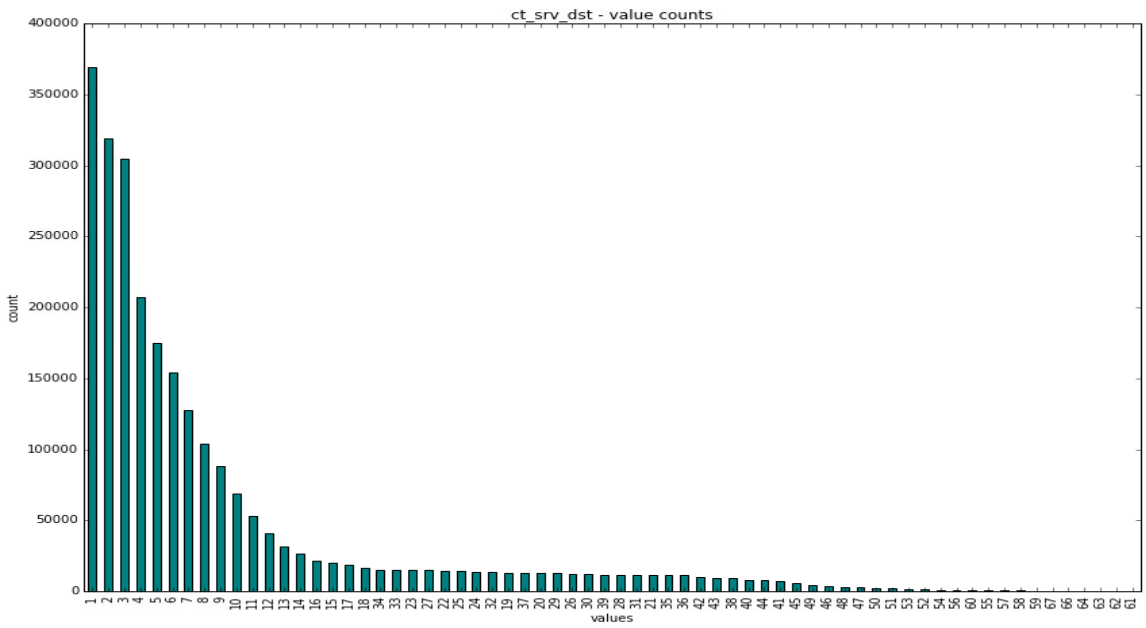
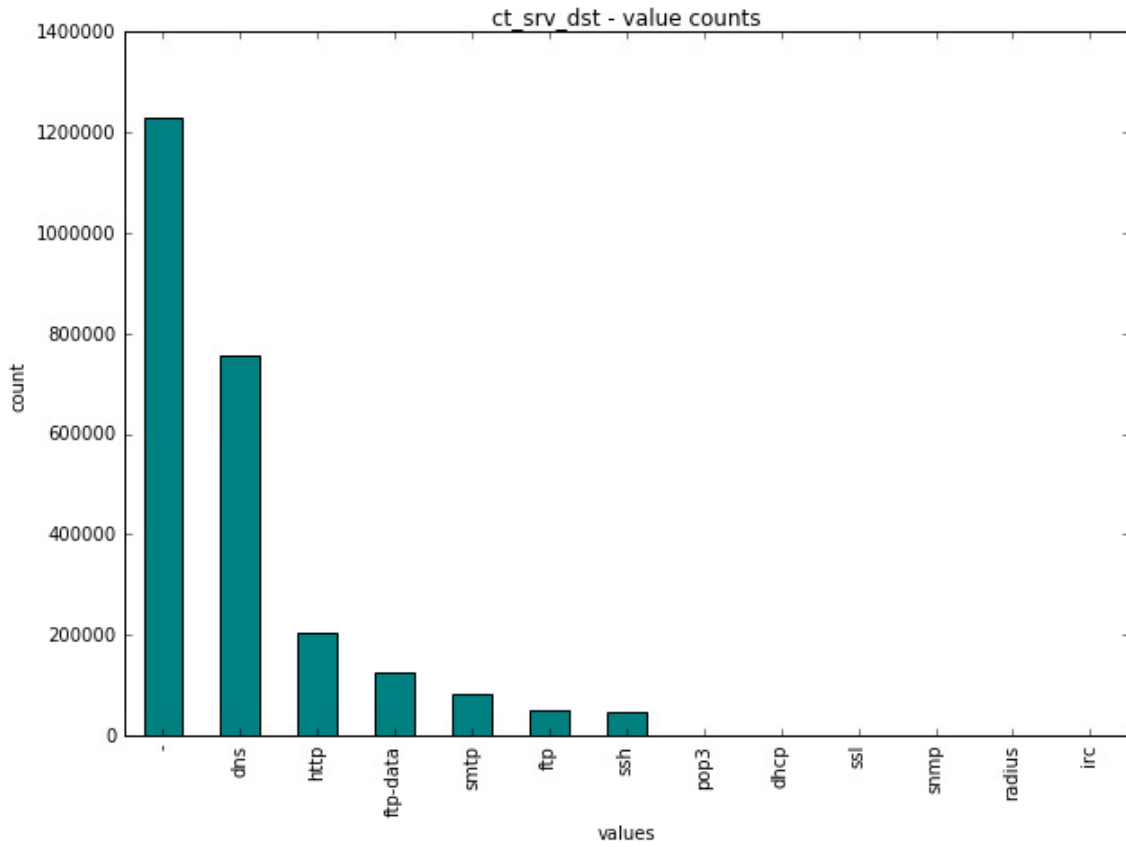
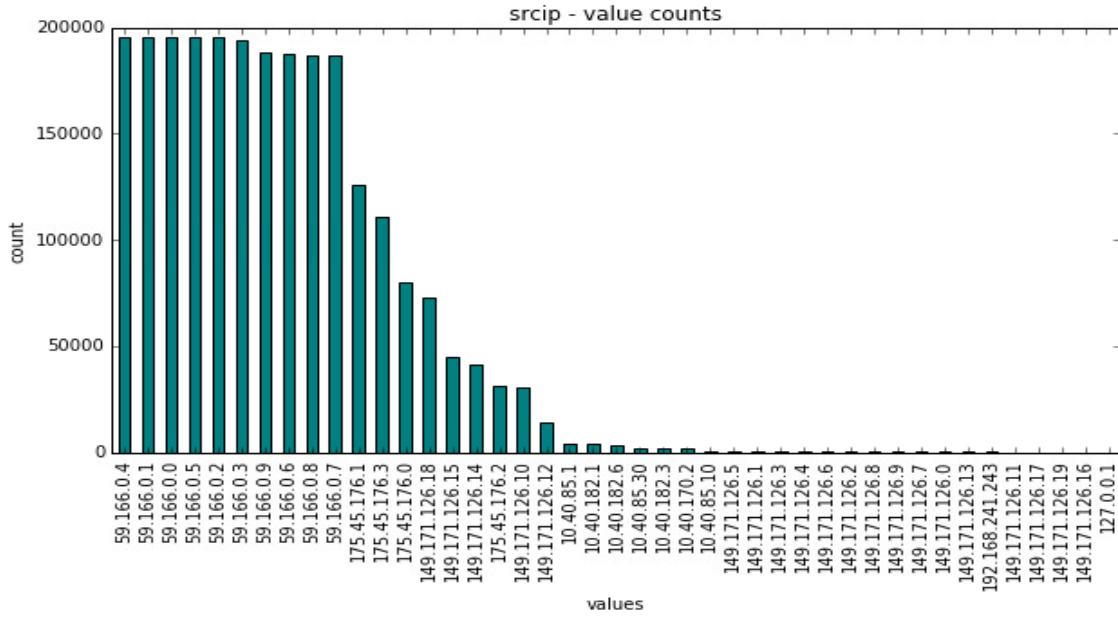


FIGURE 22: VALUE COUNT CHART - CT\_SRV\_DST

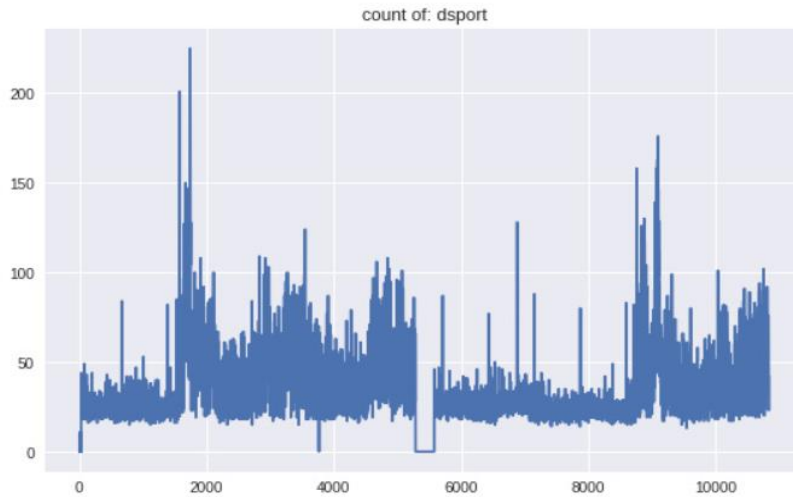


**FIGURE 23: VALUE COUNT CHART - PROTO**



**FIGURE 24: VALUE COUNTS – SRCIP (UNSW-NB15)**

Below is the count plot of the traffic in the dataset. It shows the number of packets / rows in each second. The plot is constructed from the first file of the dataset.



**FIGURE 25: COUNT OF EVENTS PER SECOND (UNSW-NB15 FILE 1)**



In Figure 26, is the scatter and density plot for some of the features within the dataset. This diagram provides us an idea how the data is distributed and how the attributes correlate amongst each other. However, such relationship between attributes will have little significance in our algorithm as we are going to club all traffic within a time-window and extract higher order features from that time-window and work only with that dataset. The diagram provides a basic high-level visibility of the attribute data space.

Scatter and Density Plot

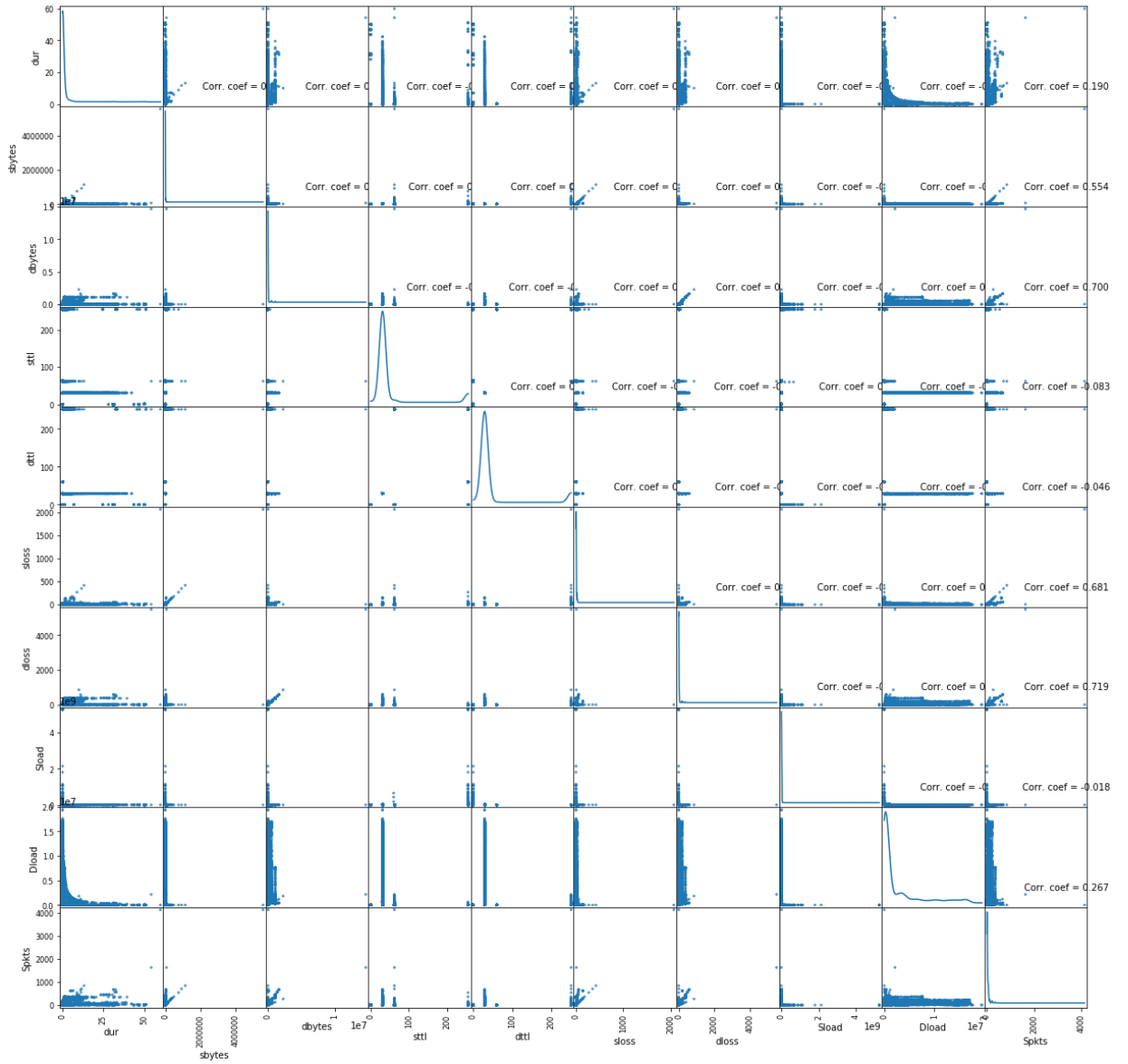


FIGURE 26: SCATTER AND DENSITY PLOT (UNSW-NB15) [72]

#### **4.5 Experiment 2: Applying a basic unsupervised clustering model on the dataset**

We would like to use a base model to attempt an unsupervised algorithm on the dataset to prove the necessity of the complex proposed algorithm. Certainly, if a simple unsupervised model can separate the attack network flows from the normal traffic, a complicated model will be deemed unnecessary. Hence, to establish the problem statement and to evaluate the results of the enhanced algorithm, the performance of the basic algorithm sets a benchmark for analysis.

We design a simple unsupervised clustering / regression model for this purpose. The entire preprocessed unlabeled traffic data is fed into the algorithm which tries to separate the data into normal and attack data in an unsupervised manner, hence without any pre-training. We feed the preprocessed dataset into the kmeans clustering implemented using scikit-learn library.

The result obtained are collected in terms of accuracy, false positive, true positive, false negative and true negative rate, precision rate, accuracy rate and F1 score. The time taken for the algorithm to work on this test setup is also measured. The results of the basic model will be presented in comparison with that of our proposed model in the section below.

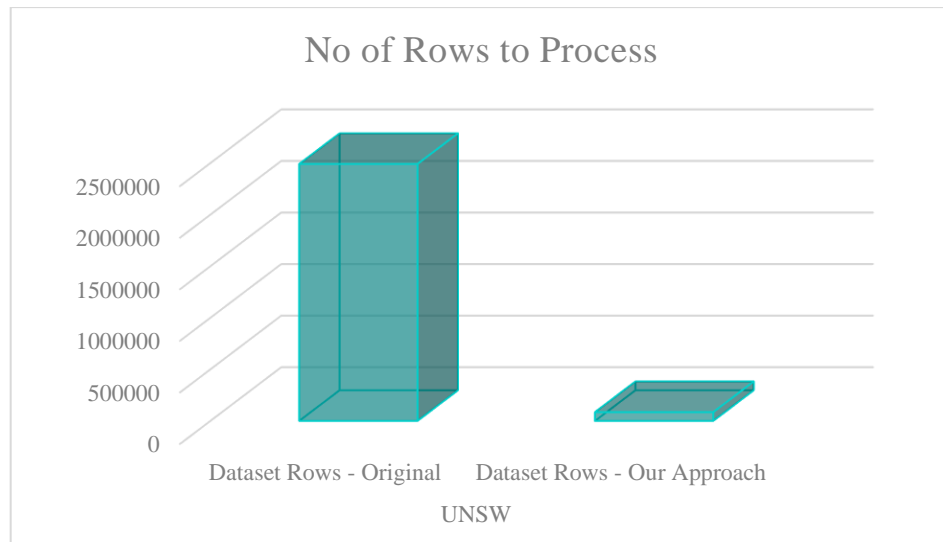
#### **4.6 Experiment 3: Execution of the Proposed Algorithm**

In the last experiment, we apply our proposed algorithm on our dataset. The entire dataset is taken and applied with the proposed unsupervised cognitive clustering, and the results including time taken for execution, and the performance metrics discussed in the

above section were stored. The results obtained from the proposed algorithm and the comparison with the traditional approaches are presented below.

#### 4.7 Results

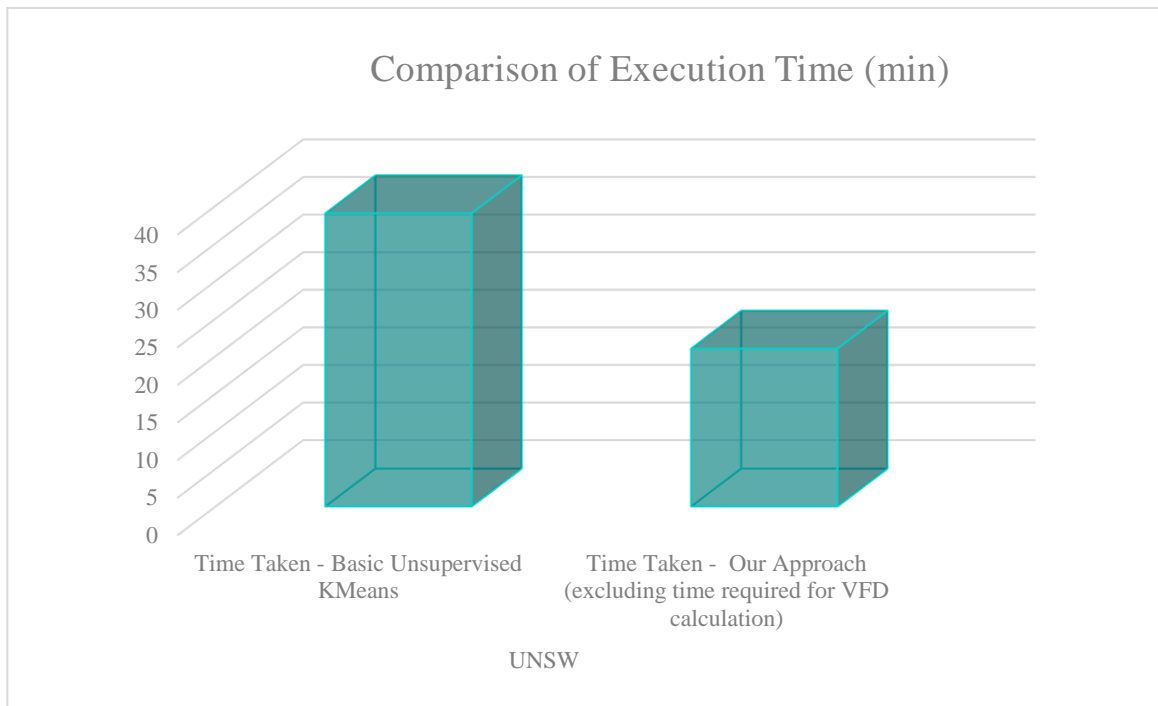
Firstly, let us assess the algorithm in terms of the number of rows that the algorithm must process, we already know that the proposed algorithm works on a much-reduced dataset. This, in itself, is an achievement, since the system now has to undergo much less processing time, and a lot less computation and storage is required to process a reduced dataset.



**FIGURE 27: COMPARISON OF NO OF DATASET ROWS TO PROCESS**

In Figure 27, the comparison is presented, with the reduced feature dataset, the dataset to be processed decreases significantly.

Now let us focus on the time taken for the execution of the algorithm. As, in our proposed method, we are dealing with a much-reduced dataset, we should be saving some execution time for the algorithm. The reduced number of rows and increased dimensionality should make it faster to bifurcate the attack and normal cloud. The below comparison is between the traditional kmeans algorithm and our approach, considering that the VFDs of the entire traffic is already calculated a priori, and in our experiment, the algorithm is only fetching the precalculated value. Hence this chart excludes the time taken to compute the VFD, which was used in the attack isolation phase. If on the contrary we include the time taken to compute the VFD, which we used in the process of attack cloud isolation, the time taken for execution would increase for our approach.



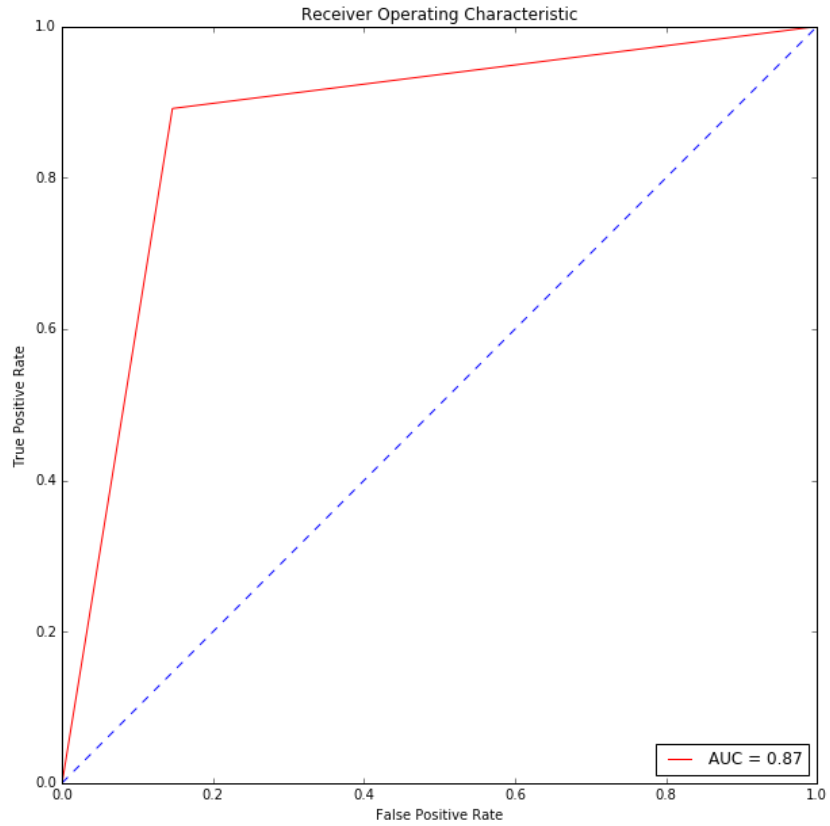
**FIGURE 28: COMPARISON OF EXECUTION TIME**

In Table 3, we summarize our results. Here we can see that the proposed algorithm has performed quite well. The basic unsupervised model values are also provided, and as we can see, even though the initial accuracy is 57% the precision, and all the metric values are low. With the proposed model, the accuracy is not as good as a supervised method, but for an unsupervised model the results are more than satisfactory. The true positive rate is 84.93, and the true negative rate is 89.65, and the accuracy is 87.27, which shows that the algorithm is able to correctly identify the normal and attack most of the times.

**TABLE 3: RESULTS OBTAINED**

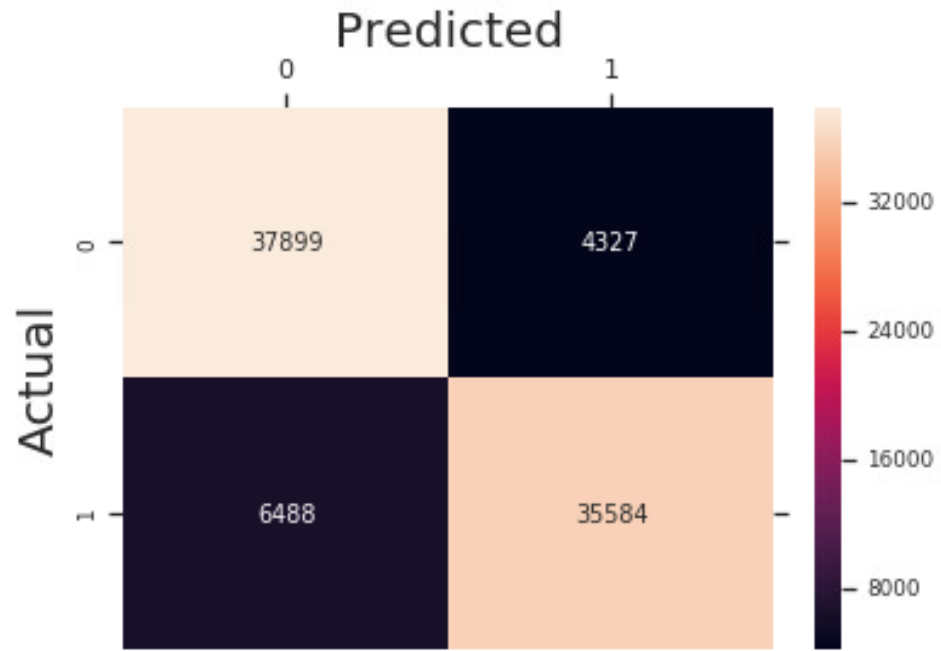
<b>Metric</b>	<b>Proposed Cognitive Unsupervised Clustering Algorithm</b>	<b>Basic unsupervised clustering</b>
<b>TPR %</b>	84.93	16.58
<b>TNR %</b>	89.65	63.75
<b>Precision %</b>	89.31	6.06
<b>Accuracy %</b>	87.27	57.92
<b>F-Measure %</b>	87.23	8.9

In Figure 29, we plot the ROC curve for the proposed algorithm. As we can see the curve is further from the 45-degree diagonal (blue dotted line in the figure) of the ROC space, and closer to the left-hand border and the top border of the ROC space which means that the algorithm is performing fairly accurately.



**FIGURE 29: ROC CURVE OF THE PROPOSED ALGORITHM**

In the below figure we plot the confusion matrix for the proposed algorithm. As can be seen, the algorithm accurately detects the attack for 36580 times, and accurately detects the normal for 37902 times. However, it also fails to detect an attack 6490 times, and falsely classifies it as a normal as an attack 4376 times.



**FIGURE 30: CONFUSION MATRIX - PROPOSED COGNITIVE UNSUPERVISED MODEL**



## Chapter 5 Conclusion and Future Work

In this section we summarize our approach and highlight on the key takeaways of the study. In this section we also discuss on the limitations and shortcoming of the study, , and later propose some promising future work and extensions of this study.

This thesis presents a novel unsupervised algorithm based on cognitive machine learning. It proposes an intelligent approach of extracting meaningful features out of unstructured, unlabeled data sources. . The algorithm combines machine learning, cognitive learning, extraction of higher-order features, and fractal-based categorization to provide a reasonable solution to the inherently complicated problem of unsupervised classification of network traffic subject to intrusion and attacks.

In contrast to the main stream research community that is more focused towards supervised learning, this study makes an attempt to explore unsupervised learning methods. Even though unsupervised learning is more complex and may seem less promising, it is important to reiterate that by giving up the training based solutions, with unsupervised learning we are making way for innovative methods that are versatile, faster and work real time.

In this thesis we have used complexity analysis for the ‘Attack Cluster Identification using VFD’ step of our algorithm (refer to

Figure 12). In this step, we randomly chose some of the items from each of the clusters, and then computed the variance fractal dimension trajectory value at the corresponding time from the original dataset. It was found that the values of the cluster of attack produced different results for the attack cloud and different results for the normal cloud, which was

used as a criteria to separate the clouds. The algorithm was able to successfully select the attack cloud, which contained the majority of the attack time steps.

In the ‘Clustering’ section of the algorithm (refer to

Figure 12), the algorithm performed k-means clustering on the unlabeled dataset of the extracted features. It is imperative that our unique feature extraction technique, wherein, we clubbed all the events in a time frame and extracted higher order features from the data streams of each attribute into a separate feature dataset, were able to extract discriminative higher order features from the dataset. The results suggest that if intelligent approaches are adapted in extracting meaningful features from unsupervised datasets, significant leverage can be achieved in classification results.

We have significantly reduced our dataset in the ‘Sampling’ and ‘Feature Extraction’ phase in our proposed model (refer to

Figure 12). In our base model we needed to perform the clustering using the entire number of rows in the dataset, however, in our proposed approach we only needed to process 84298 rows, where each of the row contained compressed information of several rows of the original dataset.

However, it is important to address that the study is performed only on one dataset. The research was performed with limited scope, time and resources. With the guidance from recommendations and future work mentioned below, there can be possible extension of this work in future beyond what has been performed already within this study.

Here are some proposed recommendations for future work:

- a) Our algorithm is sequential even though majority of the components can work in parallel and reduce the execution time further. For example, as the program is

extracting higher order features, it can fork and start extracting VFD values of randomly chosen time steps, and maintain a feedback channel to communicate with between the two instances. Eventually, as the program performs the clustering, these time-windows will fall under one or the other of the two clusters. This way the time taken for VFD calculation can be minimized. Even in the worse case scenario, when all the randomly chosen time-windows fall in the same cluster, the program only needs to go back and randomly chose some time steps from the other cluster and calculate the VFDs. Even though such improvements could not be done in this study because of limitations in time, it is worth exploring in future.

- b) Acquiring latest, real-time and effective network intrusion datasets is difficult. The study that we performed was done on one publicly available dataset, UNSW-NB15 by Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) as they have generated some of the widely popular and robust datasets. In future it can be performed on live traffic. The network model presented should be implemented on a small network in a lab, where all the network communication gets captures and written into a dataset, and then synthetically generated attacks are be introduced in the network. The algorithm should be implemented on the host and the results should be assessed.
- c) In the proposed algorithm, an kmeans clustering algorithm. Even though kmeans are very efficient in big datasets but they have several other drawbacks. In the future other advanced and deep learning algorithms can be used to cluster the clouds.

This new detection scheme may be used in other sectors like finance & economy, portfolio management, health analysis and fraud detection. Therefore, as future work,

- d) We have used our approach in the field of cybersecurity, however, this new detection scheme may be used in other sectors like finance, health analysis and fraud detection. Therefore, as future work, this proposed method can be applied in different sectors and the results can be compared to traditional approaches, and perhaps a more generalized approach in unsupervised learning can be developed with this as an underlying methodology.

## References

- [1] J. Fan, F. Han, and H. Liu, “Challenges of Big Data analysis,” *Natl. Sci. Rev.*, vol. 1, no. 2, pp. 293–314, Feb. 2014.
- [2] W. Kinsner, “It’s time for multiscale analysis and synthesis in cognitive systems,” in *Proceedings of the 10th IEEE International Conference on Cognitive Informatics and Cognitive Computing, ICCI\*CC 2011*, 2011, pp. 7–10.
- [3] R. L. Fantz, “A Method for Studying Early Visual Development,” *Percept. Mot. Skills*, vol. 6, no. 1, pp. 13–15, Mar. 1956.
- [4] A. m. Slater, “Visual perception in the newborn infant : issues and debates,” *Intellectica. Rev. l’Association pour la Rech. Cogn.*, vol. 34, no. 1, pp. 57–76, Jan. 2002.
- [5] M. Haith, “Rules that babies look by: The organization of newborn visual activity,” 1980.
- [6] G. B.-M. on infancy and undefined 1982, “The scanning patterns of human infants: Implications for visual learning,” *psycnet.apa.org*.
- [7] G. W. Bronson, “Infant Differences in Rate of Visual Encoding,” *Child Dev.*, vol. 62, no. 1, pp. 44–54, Feb. 1991.
- [8] J. R. Saffran, R. N. Aslin, and E. Newport, “Statistical learning by 8-month-old infants,” *Science (80-. )*, vol. 274, p. 1926+, Mar. 1996.
- [9] N. Moustafa and J. Slay, “UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set),” in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.
- [10] N. Moustafa and J. Slay, “The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data

- set,” *Inf. Secur. J.*, vol. 25, no. 1–3, pp. 18–31, 2016.
- [11] C. Koliás, G. Kambourakis, A. Stavrou, and S. Gritzalis, “Intrusion detection in 802.11 networks: Empirical evaluation of threats and a public dataset,” *IEEE Commun. Surv. Tutorials*, vol. 18, no. 1, pp. 184–208, 2016.
- [12] A. Patcha and J.-M. Park, “An overview of anomaly detection techniques: Existing solutions and latest technological trends,” *Comput. Networks*, vol. 51, no. 12, pp. 3448–3470, Aug. 2007.
- [13] M. H. Au, J. K. Liu, W. Susilo, and T. H. Yuen, “Secure ID-based linkable and revocable-iff-linked ring signature with constant-size construction,” *Theor. Comput. Sci.*, vol. 469, pp. 1–14, Jan. 2013.
- [14] X. Fan and G. Gong, “Accelerating signature-based broadcast authentication for wireless sensor networks,” *Ad Hoc Networks*, vol. 10, no. 4, pp. 723–736, Jun. 2012.
- [15] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld, “Toward supervised anomaly detection,” *J. Artif. Intell. Res.*, vol. 46, pp. 235–262, Feb. 2013.
- [16] M. Kuusela, T. Vatanen, E. Malmi, T. Raiko, T. Aaltonen, and Y. Nagai, “Semi-supervised anomaly detection – towards model-independent searches of new physics,” *J. Phys. Conf. Ser.*, vol. 368, no. 1, p. 012032, Jun. 2012.
- [17] R. Sommer and V. Paxson, “Outside the Closed World: On Using Machine Learning for Network Intrusion Detection,” in *2010 IEEE Symposium on Security and Privacy*, 2010, pp. 305–316.
- [18] Jiong Zhang, M. Zulkernine, and A. Haque, “Random-Forests-Based Network Intrusion Detection Systems,” *IEEE Trans. Syst. Man, Cybern. Part C (Applications Rev.)*, vol. 38, no. 5, pp. 649–659, Sep. 2008.

- [19] P. Stavroulakis and M. Stamp, Eds., *Handbook of Information and Communication Security*, 1st ed. 20. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.
- [20] H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, Jan. 2013.
- [21] C. F. Tsai, Y. F. Hsu, C. Y. Lin, and W. Y. Lin, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11994–12000, Dec-2009.
- [22] K. Leung and C. Leckie, "Unsupervised Anomaly Detection in Network Intrusion Detection Using Clusters."
- [23] W. Chimphee, A. H. Abdullah, M. N. M. Sap, S. Srinoy, and S. Chimphee, "Anomaly-based intrusion detection using fuzzy rough clustering," in *Proceedings - 2006 International Conference on Hybrid Information Technology, ICHIT 2006*, 2006, vol. 1, pp. 329–334.
- [24] M. Jianliang, S. Haikun, and B. Ling, "The application on intrusion detection based on K-means cluster algorithm," in *Proceedings - 2009 International Forum on Information Technology and Applications, IFITA 2009*, 2009, vol. 1, pp. 150–152.
- [25] S. R. Gaddam, V. V. Phoha, and K. S. Balagani, "K-Means+ID3: A novel method for supervised anomaly detection by cascading k-Means clustering and ID3 decision tree learning methods," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 3, pp. 345–354, Mar. 2007.
- [26] K. Peng, V. C. M. Leung, and Q. Huang, "Clustering Approach Based on Mini Batch Kmeans for Intrusion Detection System over Big Data," *IEEE Access*, vol. 6, pp. 11897–11906, Feb. 2018.

- [27] C.-C. Hsu and Y.-P. Huang, “Incremental clustering of mixed data based on distance hierarchy,” *Expert Syst. Appl.*, vol. 35, no. 3, pp. 1177–1185, Oct. 2008.
- [28] K. Burbeck and S. Nadjm-Tehrani, “Adaptive real-time anomaly detection with incremental clustering,” *Inf. Secur. Tech. Rep.*, vol. 12, no. 1, pp. 56–67, 2007.
- [29] L. Portnoy, L. Portnoy, E. Eskin, and S. Stolfo, “Intrusion Detection with Unlabeled Data Using Clustering,” *Proc. ACM CSS Work. DATA Min. Appl. TO Secur. (DMSA-2001)*, pp. 5--8, 2001.
- [30] K. Kim, M. E. Aminanto, and H. C. Tanuwidjaja, *Network Intrusion Detection using Deep Learning*. Singapore: Springer Singapore, 2018.
- [31] H. Motoda and H. Liu, “Feature selection, extraction and construction,” *Commun. IICM (Institute Inf. Comput. Mach. Taiwan)*, vol. 5, pp. 67–72, Jan. 2002.
- [32] E. Téglás, V. Girotto, ... M. G.-P. of the, and undefined 2007, “Intuitions of probabilities shape expectations about the future at 12 months and beyond,” *Natl. Acad Sci.*
- [33] F. Xu, V. G.-P. of the N. A. of, and undefined 2008, “Intuitive statistics by 8-month-old infants,” *Natl. Acad Sci.*
- [34] N. Z. Kirkham, J. A. Slemmer, and S. P. Johnson, “Visual statistical learning in infancy: evidence for a domain general learning mechanism.”
- [35] J. Lany, R. L. Gómez, and L. A. Gerken, “The role of prior experience in language acquisition,” *Cogn. Sci.*, vol. 31, no. 3, pp. 481–507, 2007.
- [36] F. Xu and J. B. Tenenbaum, “Word Learning as Bayesian Inference,” *psycnet.apa.org*, vol. 114, no. 2, pp. 245–272, Apr. 2007.
- [37] F. Xu and J. B. Tenenbaum, “Sensitivity to sampling in Bayesian word learning,” *Dev.*



- Sci.*, vol. 10, no. 3, pp. 288–297, May 2007.
- [38] T. Kushnir and A. Gopnik, “Conditional Probability Versus Spatial Contiguity in Causal Learning: Preschoolers Use New Contingency Evidence to Overcome Prior Spatial Assumptions,” *psycnet.apa.org*, 2007.
- [39] L. E. Schulz, E. Baraff Bonawitz, and T. L. Griffiths, “Can Being Scared Cause Tummy Aches? Naive Theories, Ambiguous Evidence, and Preschoolers’ Causal Inferences,” *Gopnik & Meltzoff*, 1995.
- [40] T. Kushnir, F. Xu, and H. M. Wellman, “Young children use statistical sampling to infer the preferences of other people.,” *Psychol. Sci.*, vol. 21, no. 8, pp. 1134–40, Aug. 2010.
- [41] A. Hyvärinen, J. Hurri, and P. O. Hoyer, *Natural Image Statistics-A Probabilistic Approach to Early Computational Vision.*, vol. 39. 2009.
- [42] “Hermann von Helmholtz - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Hermann\\_von\\_Helmholtz](https://en.wikipedia.org/wiki/Hermann_von_Helmholtz). [Accessed: 20-Mar-2020].
- [43] “Ernst Mach - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Ernst\\_Mach](https://en.wikipedia.org/wiki/Ernst_Mach). [Accessed: 20-Mar-2020].
- [44] “Karl Pearson - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Karl\\_Pearson](https://en.wikipedia.org/wiki/Karl_Pearson). [Accessed: 20-Mar-2020].
- [45] “Fergus I. M. Craik - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Fergus\\_I.\\_M.\\_Craik](https://en.wikipedia.org/wiki/Fergus_I._M._Craik). [Accessed: 20-Mar-2020].
- [46] “Edward C. Tolman - Wikipedia.” [Online]. Available: [https://en.wikipedia.org/wiki/Edward\\_C.\\_Tolman](https://en.wikipedia.org/wiki/Edward_C._Tolman). [Accessed: 20-Mar-2020].
- [47] “Carolyn Attneave - Wikipedia.” [Online]. Available:

- [https://en.wikipedia.org/wiki/Carolyn\\_Attneave](https://en.wikipedia.org/wiki/Carolyn_Attneave). [Accessed: 20-Mar-2020].
- [48] “Egon Brunswik - Wikipedia.” [Online]. Available:  
[https://en.wikipedia.org/wiki/Egon\\_Brunswik](https://en.wikipedia.org/wiki/Egon_Brunswik). [Accessed: 20-Mar-2020].
- [49] H. Barlow, “The exploitation of regularities in the environment by the brain,” *Behav. Brain Sci.*, vol. 24, no. 4, pp. 602–607, 2001.
- [50] P. Velan, M. Čermák, P. Čeleda, and M. Drašar, “A survey of methods for encrypted traffic classification and analysis,” *Int. J. Netw. Manag.*, vol. 25, no. 5, pp. 355–374, Sep. 2015.
- [51] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic, “AppScanner: Automatic fingerprinting of smartphone apps from encrypted network traffic,” in *Proceedings - 2016 IEEE European Symposium on Security and Privacy, EURO S and P 2016*, 2016, pp. 439–454.
- [52] L. Peng, B. Yang, Y. Chen, and Z. Chen, “Effectiveness of Statistical Features for Early Stage Internet Traffic Identification Deep eutectic solvents View project Effectiveness of Statistical Features for Early Stage Internet Traffic Identification,” *Artic. Int. J. Parallel Program.*, vol. 44, no. 1, pp. 181–197, Feb. 2015.
- [53] M. Xu, W. Zhu, J. Xu, and N. Zheng, “Towards selecting optimal features for flow statistical based network traffic classification,” in *17th Asia-Pacific Network Operations and Management Symposium: Managing a Very Connected World, APNOMS 2015*, 2015, pp. 479–482.
- [54] A. De Montigny-Leboeuf, “Flow attributes for use in traffic characterization,” 2005.
- [55] M. K. Korczyński and A. Duda, “Classifying Service Flows in the Encrypted Skype Traffic.”

- [56] P. Vahdani Amoli Timo Hämäläinen and G. David, “A Real Time Unsupervised NIDS for Detecting Unknown and Encrypted Network Attacks in High Speed Network.”
- [57] M. Korczyński and A. Duda, “Markov chain fingerprinting to classify encrypted traffic,” in *Proceedings - IEEE INFOCOM*, 2014, pp. 781–789.
- [58] M. S. Khan, S. Siddiqui, R. D. McLeod, K. Ferens, and W. Kinsner, “Fractal based adaptive boosting algorithm for cognitive detection of computer malware,” in *Proceedings of 2016 IEEE 15th International Conference on Cognitive Informatics and Cognitive Computing, ICCI\*CC 2016*, 2017, pp. 50–59.
- [59] H. Cohen and C. Lefebvre, *Handbook of Categorization in Cognitive Science*. Elsevier, 2005.
- [60] E. Rosch, C. B. Mervis, W. D. Gray, D. M. Johnson, and P. Boyes-Braem, “Basic objects in natural categories,” *Cogn. Psychol.*, vol. 8, no. 3, pp. 382–439, Jul. 1976.
- [61] W. Kinsner, “Complexity and its measures in cognitive and other complex systems,” *Proc. 7th IEEE Int. Conf. Cogn. Informatics, ICCI 2008*, pp. 13–29, 2008.
- [62] E. D. Reilly, A. Ralston, and D. Hemmendinger, *Encyclopedia of computer science*. Wiley, 2003.
- [63] E. Lindenstrauss and B. Weiss, “Mean topological dimension,” *Isr. J. Math.*, vol. 115, no. 1, pp. 1–24, 2000.
- [64] W. Kinsner, “A unified approach to fractal dimensions,” in *Fourth IEEE Conference on Cognitive Informatics 2005, ICCI 2005*, 2005, pp. 58–72.
- [65] L. Zhao, W. Li, L. Geng, and Y. Ma, “Artificial neural networks based on fractal growth,” in *Lecture Notes in Electrical Engineering*, 2011, vol. 123 LNEE, pp. 323–330.

- [66] E. Bieberich, “Recurrent fractal neural networks: a strategy for the exchange of local and global information processing in the brain.”
- [67] E. Kim, M. Sano, Y. S.-P. of theoretical physics, and undefined 1993, “Fractal neural network: Computational performance as an associative memory,” *academic.oup.com*.
- [68] S. Siddiqui, M. S. Khan, K. Ferens, and W. Kinsner, “Detecting advanced persistent threats using fractal dimension based machine learning classification,” in *IWSPA 2016 - Proceedings of the 2016 ACM International Workshop on Security and Privacy Analytics, co-located with CODASPY 2016*, 2016, pp. 64–69.
- [69] W. McKinney, “Data Structures for Statistical Computing in Python,” in *Proceedings of the 9th Python in Science Conference*, 2010, pp. 56–61.
- [70] F. Pedregosa FABIANPEDREGOSA *et al.*, “Scikit-learn: Machine Learning in Python Gaël Varoquaux Bertrand Thirion Vincent Dubourg Alexandre Passos PEDREGOSA, VAROQUAUX, GRAMFORT ET AL. Matthieu Perrot,” 2011.
- [71] K. Nahiyani, S. Kaiser, K. Ferens, R. McLeod, { Nahiyank, and }@myumanitoba Ca, “A Multi-agent Based Cognitive Approach to Unsupervised Feature Extraction and Classification for Network Intrusion Detection.”
- [72] “Starter: UNSW\_NB15 523d3e7e-a | Kaggle.” [Online]. Available: <https://www.kaggle.com/kerneler/starter-unsw-nb15-523d3e7e-a>. [Accessed: 19-Feb-2020].
- [73] “The UNSW-NB15 data set description.” [Online]. Available: <https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/>. [Accessed: 17-Apr-2020].

## APPENDIX A - List of the features of UNSW-NB15 Dataset

Referring to [9] [73]

#	Name	Name Description
<b>1. Flow Features</b>		
1	srcip	Source IP address.
2	sport	Source port number.
3	dstip	Destinations IP address
4	dsport	Destination port number.
5	proto	Protocol type, such as TCP, UDP.
<b>2. Basic Features</b>		
6	state	The states and its dependent protocol e.g., CON.
7	dur	Row total duration.
8	sbytes	Source to destination bytes.
9	dbytes	Destination to source bytes.
10	sttl	Source to destination time to live.
11	dttl	Destination to source time to live.
12	sloss	Source packets retransmitted or dropped.
13	dloss	Destination packets retransmitted or dropped
14	service	Such as http, ftp, smtp, ssh, dns and ftp-data.
15	sload	Source bits per second.
16	dload	Destination bits per second.
17	spkts	Source to destination packet count.
18	dpkts	Destination to source packet count.
<b>3. Content Features</b>		
19	swin	Source TCP window advertisement value.

20	dwin	Destination TCP window advertisement value.
21	Stcpb	Source TCP base sequence number.
22	dtcpb	Destination TCP base sequence number.
23	smeansz	Mean of the packet size transmitted by the srcip.
24	dmeansz	Mean of the packet size transmitted by the dstip.
25	trans_depth	The connection of http request/response transaction.
26	res_bdy_len	The content size of the data transferred from http.
<b>4. Time Features</b>		
27	sjit	Source jitter.
28	djit	Destination jitter.
29	stime	Row start time.
30	ltime	Row last time.
31	sintpkt	Source inter-packet arrival time.
32	dintpkt	Destination inter-packet arrival time.
33	tcprtt	Setup round-trip time, the sum of 'synack' and 'ackdat'.
34	synack	The time between the SYN and the SYN_ACK packets.
35	ackdat	The time between the SYN_ACK and the ACK packets.
36	is_sm_ips_ports	If srcip (1) = dstip (3) and sport (2) = dsport (4), assign 1 else 0.
<b>5. Additional Generated Features</b>		
37	ct_state_ttl	No. of each state (6) according to values of sttl (10) and dttl (11).
38	ct_flw_http_mthd	No. of methods such as Get and Post in http service.
39	is_ftp_login	If the ftp session is accessed by user and password then 1 else 0.
40	ct_ftp_cmd	No of flows that has a command in ftp session.

41	ct_srv_src	No. of rows of the same service (14) and srcip (1) in 100 rows.
42	ct_srv_dst	No. of rows of the same service (14) and dstip (3) in 100 rows.
43	ct_dst_ltm	No. of rows of the same dstip (3) in 100 rows.
44	ct_src_ltm	No. of rows of the srcip (1) in 100 rows.
45	ct_src_dport_ltm	No of rows of the same srcip (1) and the dsport (4) in 100 rows.
46	ct_dst_sport_ltm	No of rows of the same dstip (3) and the sport (2) in 100 rows.
47	ct_dst_src_ltm	No of rows of the same srcip (1) and the dstip (3) in 100 records.
<b>6. Labelled Features</b>		
48	Attack_cat	The name of each attack category.
49	Label	0 for normal and 1 for attack records