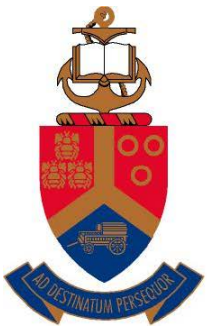


UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Cybersecurity: the intelligent discovery of malicious bots

by

Innocent Mbona

Submitted in fulfilment of the requirements for the degree

Doctor of Philosophy (Information Technology)

in the Department of Computer Science

Faculty of Engineering, Built Environment and Information Technology (EBIT)

University of Pretoria

Supervisor: Prof J.H.P Eloff

Pretoria (South Africa)

December 2024

Thesis summary

Cybersecurity: the intelligent discovery of malicious bots

by

Innocent Mbona

E-mail: u15256422@tuks.co.za

Supervisor : Prof J.H.P Eloff

Department : Computer Science

University : University of Pretoria

Degree : Doctor of Philosophy

Keywords : Cybersecurity, cyberattack, cyber threat intelligence, anomaly detection, bots, Benford's law, feature selection, machine learning, Internet Application Platforms

Abstract

Computer programs known as bots are increasingly affecting various aspects of our lives, including entertainment and technology. Bots have been created by humans to perform particular activities on Internet Application Platforms (IAPs), for instance, to search for information via search engines. Unfortunately, the capabilities of bots can be hijacked by cybercriminals, who use them to launch cyberattacks. The attacks launched by bots on IAPs impose serious cybersecurity risks, as bots are able to launch large-scale cyberattacks that might be difficult to discover. Thus, it is important that IAPs implement intelligent decision-making cybersecurity solutions that can combat such cyberattacks.

Machine learning (ML) algorithms enable the design of intelligent decision-making cybersecurity solutions. The foremost step in the design of ML-based cybersecurity solutions is data

preparation, including feature selection. Due to the variety of users found on IAPs - such as computer networks with a large number of features, coupled with high volumes of data that evolve continuously - the existing feature selection methods face serious challenges. Although feature selection methods such as the principal component analysis (PCA) perform well on high-dimensional imbalanced data problems, they can be computationally expensive. Thus, effective and easy-to-implement feature selection methods are required for large-volume, high-dimensional, imbalanced data problems.

This thesis proposes a methodological approach named CySecML, which provides a framework for developing intelligent ML-based cybersecurity solutions that can assist cyber threat intelligence (CTI) procedures to effectively discover cyber threats launched by bots on IAPs. The CySecML methodology is based on two components - data preparation and the InternetBotDetector model, as it aims to optimise existing techniques that include data quality checks, feature selection and ML on cybersecurity data sets. To provide proof-of-concept of this methodology, two different IAPs namely - online social networks (OSNs) and network intrusion detection systems (NIDSs) were chosen to discover bot cyberattacks.

Herein, this thesis's original and significant contribution to existing knowledge within the field of cybersecurity stem from the following contributions: (i) Proposes a unique CySecML methodology for developing intelligent ML-based cybersecurity solutions, (ii) Proposes an efficient and easy-to-implement feature selection method through Benford's law on cybersecurity data sets, (iii) Proposes a bot taxonomy from a cybersecurity perspective, and (iv) Contributes towards the design of intelligent solutions for discovering ever-growing attacks such as zero-day network intrusion and social engineering attacks.

Declaration and Ethical Clearance

Declaration

I, the undersigned hereby declare that:

- I understand what plagiarism is and I am aware of the University's policy in this regard;
- The work contained in this thesis is my own original work;
- I did not refer to work of current or previous students, lecture notes, handbooks or any other study material without proper referencing;
- Where other people's work has been used this has been properly acknowledged and referenced;
- I have not allowed anyone to copy any part of my thesis;
- I have not previously in its entirety or in part submitted this thesis at any university for a degree.

Disclaimer

The work presented in this thesis is the original work of the author of this thesis. The author of this thesis was responsible for all the technical aspects of the research including developing test scenarios, conducting experiments, analysis, observations, findings, and writing of the thesis with limited guidance from the supervisor.

Ethics statement

The author whose name appears on the title of this thesis has obtained - for the research described in this work - the applicable research ethics approval.

The author declares that he has observed the ethical standards required in terms of the University of Pretoria's Code of Ethics for Researchers and Policy guidelines for responsible research.

Full names of student: Innocent Nqoba Mbona

Student number: u15256422

Date submitted: December 2024

Student's signature:

Supervisor's signature:

Acknowledgements

I give glory and praise to God for His goodness and mercy over my life.

I would like to thank the following people and organisations who played a key role in assisting me to complete my doctoral studies.

- My sincere gratitude to my supervisor Prof J.H.P Eloff for all the time and resources he devoted towards my doctoral studies. I am fortunate to have learned so much from him.
- I am forever indebted to my mother and brother Skhumbuzo Aubrey Mbona who were responsible for my upbringing and shaping my educational career path.
- To my fiancée Katlego, thank you for your unwavering love, support and patience.
- I thank my family, friends, and colleagues for your encouragement and support throughout my studies.
- I am grateful to Bankseta for providing financial support towards my studies.
- A special thanks to Isabel Claassen for applying her language expertise to edit this thesis.

Publications and contributions

Throughout this thesis, the main findings and contributions have been published in the journals and conference proceedings mentioned below. The articles mentioned below are my own original work and incorporated in this thesis.

Peer-reviewed journal articles

I. Mbona and J.H.P. Eloff, "Feature selection using Benford's law to support detection of malicious social media bots", Information Sciences, vol. 582, pp. 369-381, Sep 2021

Contribution: The main contribution of this article was the proposal of Benford's law (BL) as a novel feature selection method. BL was used to identify meaningful features indicative of anomalous behaviours between humans and malicious OSN bots. This article also demonstrated that Benford's law is an effective method for distinguishing between human and malicious bot activities in OSNs. Furthermore, meaningful features identified by Benford's law were consistent with the results obtained by the principal component analysis and ensemble random forest methods on the same data sets. This article contributes towards the design of intelligent machine learning based cybersecurity solutions for discovering bot cyberattacks in OSNs.

Where and how incorporated in the thesis: this article was incorporated in Chapter 7 to motivate and describe the use of Benford's law as a feature selection method on OSN cybersecurity data sets.

I. Mbona and J.H.P. Eloff, "Detecting Zero-Day Intrusion Attacks Using Semi-Supervised Machine Learning Approaches", IEEE Access, vol. 10, pp. 69822-69838, Jul 2022

Contribution: This article contributed towards addressing the ever-challenging security issue of discovering zero-day network intrusion attacks. The contribution was twofold: (i) It proposed using Benford's law as a network traffic analysis technique for differentiating between benign and zero-day network intrusion attacks; (ii) Owing to the practical limitation of obtaining large amounts of labelled data sets to train network intrusion detection systems (NIDSs) to combat zero-day attacks, this article demonstrated that semi-supervised machine learning (SSML) can be effective in discovering zero-day network intrusion attacks. SSML require small sample of labelled and large amounts of unlabelled data points to train a machine learning algorithm.

Where and how incorporated in the thesis: this article was incorporated in Chapters 7 and 8 to describe significant features and semi-supervised machine learning algorithms respectively, used in this thesis to discover zero-day intrusion attacks.

I. Mbona and J.H.P. Eloff, "Classifying social media bots as malicious or benign using semi-supervised machine learning", Journal of Cybersecurity, vol. 9, Issue 1, Jan 2023

Contribution: The main contribution of this article involved analysing the behavioural features of benign and malicious bots found in OSNs. At the time, the majority of existing studies focused on identifying meaningful features that assist in differentiating between humans and malicious bots, and little research was done to differentiate between benign and malicious bots. From a cybersecurity perspective, the nature of malicious bots differs from that of benign bots in that benign bots perform useful activities, whereas malicious bots do not. This article adopted Benford's law to identify meaningful features indicative of anomalous behaviour between benign and malicious bots. The effectiveness of this approach was demonstrated by evaluating semi-supervised machine learning algorithms to classify benign and malicious bots. One of the main findings of this article was that meaningful features that have been used successfully in the literature to classify humans and malicious bots, are not equally successful in classifying

benign and malicious bots based on the experiments conducted.

Where and how incorporated in the thesis: this article was incorporated in Chapters 7 and 8 to describe significant features and semi-supervised machine learning algorithms respectively, used in this thesis to discover social engineering attacks.

Peer-reviewed conference articles

At the time of writing this thesis, the first two conference proceedings were accepted for publication at the Computer Science, Computer Engineering and Applied Computing (CSCE). However, due to the Coronavirus (COVID-19) pandemic, there have been delays in publishing proceedings; therefore, these articles do not have citations.

I. Mbona and J.H.P. Eloff, "Evaluating a semi-supervised intrusion detection algorithm through Benford's law"

Conference details: The 2021 World Congress in Computer Science, Computer Engineering, and Applied Computing. July 26-29, 2021 Las Vegas, USA. Conference proceedings from the 17th International Conference on Data Science (ICDATA 2021).

Book of abstract is found here: https://www.american-cse.org/static/CSCE21%20book%20abstracts.pdf , page 106[1].

Contribution: Network traffic data is typified by imbalanced and high-dimensions, for example, low observations of network intrusion attacks and high observations of benign network traffic. This article addresses the problem of analysing network traffic data in real-time using Benford's law. To demonstrate the effectiveness of this approach, significant features of denial-of-service (DoS) and brute-force attacks were investigated using Benford's law, information gain, principal component analysis (PCA), synthetic minority oversampling technique (SMOTE), ensemble feature selection, Chi-squared, XGBoost, CatBoost, random forest and Light Gradient Boosting Machine (LightGBM) as feature selection methods, and they were evaluated using the

[1]Last accessed: 01May2024.

semi-supervised Gaussian mixture model. This work was further extended in the paper entitled: "Detecting zero-day intrusion attacks using semi-supervised machine learning approaches".

Where and how incorporated in the thesis: this article was incorporated in Chapter 7 to motivate and describe the use of Benford's law as a feature selection method on NIDSs cyber-security data sets.

I. Mbona and J.H.P. Eloff, "Taxonomy of bots from a Cybersecurity perspective"

Conference details: The 2022 World Congress in Computer Science, Computer Engineering, and Applied Computing. July 25-28, 2022 Las Vegas, USA. Conference proceedings from the 21st International Conference on Security and Management (SAM 2022).

Book of abstract is found here: https://www.american-cse.org/static/CSCE22-book-abstracts-printing.pdf , page 147[2].

Contribution: This article contributed in that it conducted a systematic literature review based on a pre-defined search strategy, databases, and inclusion or exclusion criteria. This systematic literature review study was conducted in accordance with the preferred reporting items for systematic reviews and meta-analyses (PRISMA) framework. The aim was to provide a high-level overview of the cybercrimes launched by bots. This was achieved by proposing a bot taxonomy based on bot cybercrime, bot type, bot domain, and detection or prevention methods. Moreover, research gaps such as the accurate discovery of new unknown attacks launched by bots were identified and possible future directions were discussed.

Where and how incorporated in the thesis: this article was incorporated in Chapter 2 and 3, as the systematic literature review and related work respectively.

[2]Last accessed: 01May2024.

I. Mbona and J.H.P. Eloff, "Data Sets for Cyber security Machine learning models: a Methodological Approach", Proceedings of the 9th International Conference on Internet of Things, Big Data and Security (IoTBDS), pp. 149-156, 2024 , Angers, France

Conference details: 9th International Conference on Internet of Things, Big Data and Security (IoTBDS 2024). Angers, France, 28-30 April 2024.

Book of abstract is found here: https://iotbds.scitevents.org/Abstract.aspx?idEvent=Jp8TXU9uxEQ= , page 3[3].

Contribution: The contribution of this article is its proposal of a methodological approach that can assist organisations in developing intelligent decision-making machine learning based cybersecurity solutions. This methodological approach is called CySecML and consist of two key components that are data preparation and InternetBotDetector model. The data preparation component provides guidelines for obtaining cybersecurity data, checking data quality and feature selection. The InternetBotDetector model component is a machine learning based model used to intelligently discover cyberattacks.

Where and how incorporated in the thesis: this article was incorporated in Chapters 4,5 and 6 to motivate, describe the development and the design of the CySecML methodology.

[3]Last accessed: 05May2024.

List of Acronyms

The following list provides the details of the acronyms that were used in this thesis.

Acronym	**Description**
OSNs	Online Social Networks.
$\mathbb{X}$	OSN platform formally known as Twitter.
NIDSs	Network Intrusion Detection Systems.
IoT	Internet of Things.
IoE	Internet of Everything.
PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses.
ML	Machine Learning.
UML	Unified Modeling Language.
API	Application Programming Interface.
URL	Uniform Resource Locator.
NN	Neural Network.
SVM	Support Vector Machine.
PCA	Principal Component Analysis.
WWW	World Wide Web (Web for short).
F_1 score	$0 \leq F_1 \leq 1$ is the *harmonic* mean of the precision and recall.
AUC	Area under the curve.
$\mathcal{F}$	$\sigma-$algebra.
Ω	Probability space.
$\mathbb{P}$	Probability measure.
$\mathbb{E}$	Expectation operator under some measure.
MCC score	Matthews Correlation Coefficient.

Table 1: Acronyms and Descriptions

Acronym	Description
min	Minimum.
max	Maximum.
std	Standard deviation.
IAT	Inter-arrival time.
avg	Average.
BL	Benford's law.
SSML	Semi-supervised machine learning.
DNS	Domain Name System.
IP	Internet Protocol.
LDAP	Lightweight Directory Access Protocol.
MSSQL	Microsoft SQL Server.
NetBIOS	Network Basic Input/Output System.
NTP	Network Time Protocol.
TCP	Transmission Control Protocol.
SNMP	Simple Network Management Protocol.
SSDP	Simple Service Discovery Protocol.
UDP	User Datagram Protocol.
TFTP	Trivial File Transfer Protocol.
HTTP	Hypertext Transfer Protocol.
CTI	Cyber Threat Intelligence.
CIA	Confidentiality Integrity Availability.
DoS	Denial-of-service.
DDoS	Distributed-denial-of-service.

Table 2: (*Continued.*) Acronyms and Descriptions

Glossary

This chapter provides definitions and terminologies used in this thesis.

- True Positive (TP): cyber activity that is predicted to be malicious, and it is malicious.
- True Negative (TN): cyber activity that is predicted to be benign, and it is benign.
- False Positive (FP): cyber activity that is predicted to be malicious, but it is benign.
- False Negative (FN): cyber activity that is predicted to be benign, but it is malicious.
- Confusion matrix is used to determine the performance of a machine learning algorithm.

	Actual malicious	**Actual benign**
Predicted malicious	TP	FP
Predicted benign	FN	TN

Table 3: Confusion matrix

From the confusion matrix in Table 3, one can derive the accuracy, precision, recall, F_1 score and MCC score performance measures.

- Accuracy: the number of true positive and true negative results as a proportion of total observation.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision: the number of true positive results as a proportion of positive results.

$$Precision = \frac{TP}{TP + FP}$$

- Recall: the true positive rate.

$$Recall = \frac{TP}{TP + FN}$$

- F_1 score: the harmonic mean of precision and recall.

$$F_1 = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} = 2\frac{Precision * Recall}{Precision + Recall}$$

- MCC score: the measure of correlation between predicted and verified cases.

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

- An attribute: In this thesis, an attribute is defined as a data container for information describing the behaviour of a system [1, 2, 3]. For example, in a network intrusion detection system, *packets* that contain original data or flow of information are considered as an attribute.

- A feature: In this thesis, a feature is defined as information that can be extracted or derived from an attribute [1, 2, 3]. For example, using packets, it is possible to derive further information about the size of packets, which is a feature called packet size. However, in this thesis features and attributes are terms that were used interchangeably without any impact.

- A significant or meaningful feature: In this thesis, a significant or meaningful feature is defined as information that can distinctively differentiate between two classes in a data set.

- Threat discovery versus threat detection: These terminologies are used interchangeably in the literature without any impact, although there is a subtle difference. The author of this thesis defines discovery as a process of finding something, especially previously not known, whereas detection is the process of finding something known. For example, finding an unknown threat can be referred to as discovering, and finding a known threat can be referred to as detecting.

Contents

List of Figures

List of Tables

Introduction

Chapter 1

Introduction

1.1 Background

The Internet is considered a major innovation that enables humans to interact in real-time through devices such as computers and smartphones, irrespective of geographic location [14, 15]. The platform that enables virtual interaction is called cyberspace, and it uses the Internet (among other things) for people to interact and exchange information [16, 17]. People use the Internet for various reasons, including searching for information on the worldwide web (Web) and connecting with friends via online social networks (OSNs) platforms [17, 18]. According to Kremling et al. [16], cyberspace rests on four main elements, namely physical - computers or other devices; logical - the design of the Internet; information - sharing of data; and personal - building Web pages, and so on. The use of the Internet has significantly increased over the last number of decades, giving rise to new and exciting technologies such as the Internet of Things (IoT) and the Internet of Everything (IoE) [19], which connect various elements of cyberspace into a single environment.

IoT refers to smart objects or “things” such as smart watches and phones that are interconnected using methods that include unique addressing schemes and standard communication protocols, whereas the IoE refers to more than “things” that are interconnected [19]. According to Miraz et al. [19], IoE comprises four main components: things, people, data, and processes. The rapid evolution of technologies such as the IoT and IoE has increased the demand for the development of bots that improve the automation of activities such as the exchange of data [20, 21, 22]

within the IoT and IoE domains. IoT and IoE devices often communicate or exchange information through bots, which play an important role in automating the workflow among IoT devices or end users [23, 24]. For example, Mardini et al. [20] developed a bot that connects a Facebook messenger with IoT devices using an application programming interface (API). This messenger bot enables users to send and receive Facebook messages on their IoT devices such as smart television. Kar et al. [25] proposed integrating chat bots to IoT devices such as smart cars and smart homes using the Web API, hypertext transfer protocol (HTTP) and representational state transfer (REST) architectures.

The above examples are cases of benign bots that are used for legitimate purposes. In contrast, Chatterjee et al. [23] demonstrated how cybercriminals can exploit unsecured IoT devices within IoT networks to execute malicious activities. A well-known case study of malicious bot attack in the IoT environment is the Mirai botnet attack that occurred in 2016 [24, 26]. Ever since the Mirai botnet attack, researchers and practitioners have proposed various methods of discovering and preventing this constantly growing cybersecurity threat [24, 27]. It is crucial to discover cyber threats posed by malicious bots in cyberspace, as bots are capable of launching large-scale cyberattacks that are difficult to discover. Various types of bots have been developed for different purposes, as was illustrated by the aforementioned examples.

The term bot is derived from “robot”, and several definitions for bots have been reported in the literature [28, 29]. Their definitions differ depending on the context in which they are used [30, 28, 31]. For example, most authors define a bot as a computer program that seeks to mimic human behaviour on the Web to avoid detection [21, 32], whereas others define it as a harmful computer program that seeks to mislead humans on the Web [33, 34]. In this thesis, a bot is defined as any software application that can execute automated tasks on the Internet. Bots can be used for various activities on the Internet, and some practical examples have already been discussed in this introduction section. Most bot activities are benign, e.g., those used in smart homes to set temperatures, whereas others, e.g., Web scraping which are used to illegally collect information on the Web, are malicious and pose serious cybersecurity threats that often lead to lawsuits and reputational damage for organisations [18, 32, 35].

Cybersecurity is derived from cyberspace and refers to the protection of electronic data and computers against threats, as well as to the confidentiality, integrity, and availability (CIA) of the systems [16, 36, 37]. In this thesis, any malicious activity by bots in cyberspace is consid-

ered as a cybercrime or cyberattack. A cybercrime or cyberattack can be defined as a criminal or malicious activity that occurs over the Internet through the use of smart devices [16, 37]. This thesis specifically investigates cyber threats or attacks posed by bots in cyberspace. Figure 1.1 illustrates the research focus of this thesis, as indicated by the dotted red lines.

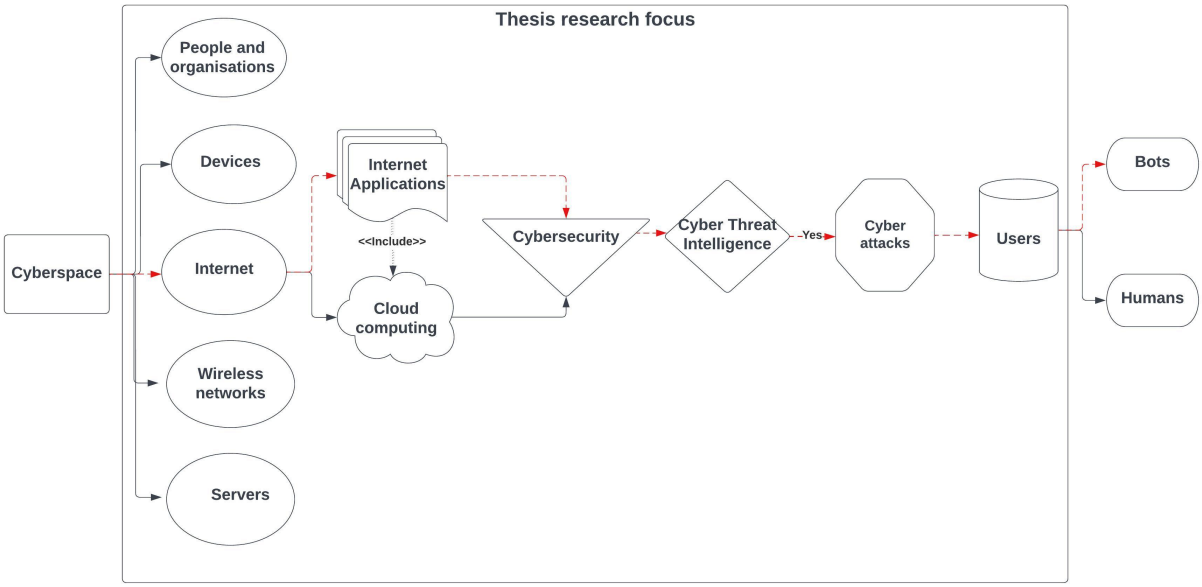


Figure 1.1: Research focus area covered by this thesis

The literature reflects various views on the elements that constitute a cyberspace [16, 17, 38], and Figure 1.1 highlights the most common elements: devices, wireless networks, people and organisations, servers, and the Internet. The two main components of the Internet are cloud computing - the technology that enables Internet users to store and access information from any device - and Internet applications - an umbrella term that describes the application platforms of the Internet. The latter include search engines, Web browsers, e-commerce, online payments, and OSNs among others. Internet Application Platforms (IAPs) are protected from cyberattacks by cyber threat intelligence (CTI) procedures that form part of cybersecurity measures [39, 40]. A CTI procedure is a set of guidelines that enable system controllers to proactively make informed decisions to discover, analyse, and mitigate malicious activities on the Internet [41, 42]. Discovering cyberattacks in real-time is one of the ultimate cybersecurity goals, and CTI procedures play a crucial role in this regard [41, 42]. cyberattacks on IAPs can be carried out by humans or bots [38]. In this thesis, the focus is limited to cyberattacks launched by bots, as they have the capability to execute large-scale cyberattacks much faster than humans can do [29, 38].

Some of the many real-world cases of cyberattacks launched by malicious bots include social engineering attacks [43, 44, 45], cyber-bullying [17, 46, 47], DDoS [36, 48, 49] and dissemination of fake news [50, 51]. The dissemination of fake news on OSNs using bots is a serious cyberattack affecting many people's lives. For example, it is believed that bots influenced the Venezuela elections in 2015 through the dissemination of fake news [52, 53], and similar cases were later observed in the presidential elections of the United States of America in 2016 [54, 55] and Swedish general elections in 2018 [56]. Cybercriminals are continuously evolving their attack strategies, which has led to rising numbers of new unknown attacks referred to as zero-day attacks [57]. Such attacks can cause devastating financial losses to organisations and individuals [57, 37]. A data breach can cause an organisation enormous amounts of damage. For instance, in the United States of America, a data breach can cost on average approximately USD 4 million annually, estimated for the period 2016 - 2020 [37, 58]. Wireless networks are particularly vulnerable to cyberattacks because the defence system may not be fully prepared to defend itself, and cybercriminals execute these attacks using very sophisticated methods [59, 60].

The main defence strategy to safeguard against cyberattacks is to adopt cybersecurity frameworks such as ISO 27001 or ISO 27002 [61, 62] and the National Institute of Standards and Technology (NIST) [61, 63]. Such frameworks assist organisations to have a strong vulnerability management system. This implies that they are clear about their threat analysis, their vulnerability analysis, and their response strategies to threats such as cyberattacks launched by bots. Any cybersecurity framework rests on five main pillars: identify, protect, discover, respond, and recover [36, 64]. In this thesis, the main focus is on the discovery pillar of cybersecurity frameworks.

Over the years, considerable research has been conducted to enhance the discovery pillar of cybersecurity frameworks, particularly for cyberattacks launched by malicious bots. State-of-the-art research demonstrates that cybersecurity solutions based on machine learning (ML) can be used to intelligently discover malicious bot cyberattacks [65, 66, 67]. For example, researchers [7, 68, 69] adopted ML-based cybersecurity solutions to discover phishing attacks launched by malicious bots, and [70, 71, 72] to discover network intrusion attacks launched by bots. When designing ML algorithms for discovering malicious bot cyberattacks, input features and availability of labelled attack data sets are key factors for consideration on big data platforms [73, 74]. For instance, IAPs such as OSNs and NIDSs are considered big data platforms, given the variety, velocity, and volume of data generated from these platforms [75, 76].

Specifically for this thesis, the question remains: as cybercriminals are employing sophisticated complex cyberattacks such as zero-day network intrusion attacks, what are the optimal set features that should be monitored to intelligently discover cyberattacks launched by malicious bots in big data platforms? Moreover, how can ML-based cybersecurity solutions be enhanced to improve the decision-making of discovering cyberattacks launched by bots? The current thesis aimed to address precisely these questions.

1.2 Study motivation

Recently, there has been an increase in the development of bots driven mainly by exciting new technologies such as the IoT and IoE. Bots are generally used for legitimate purposes on Internet Application Platforms (IAPs); however, there have been cases in the past where they were used to execute large-scale cyberattacks. Cybercriminals are continuously deploying sophisticated bots on IAPs that are difficult to discover. cyberattacks launched by bots on IAPs cause not only devastating financial losses but also compliance breaches that can damage the reputation of organisations and individuals. Therefore, there is a need to do further research to understand bots from a cybersecurity perspective and enhance CTI procedures with intelligent decision-making cybersecurity solutions to protect humans and cyber infrastructures from malicious bots, as this is of utmost importance from a cybersecurity perspective.

The CTI procedure includes the extraction of data from IAPs such as traffic logs, processing of this raw data into a readable format suitable for analysis, and thereafter, the analysis and discovery of cyber threats using intelligent tools such as machine learning [77, 40, 78]. The data extracted from the IAPs are generally represented by features that describe various components of IAPs. For example, user behaviour can be described by features such as user-based, network, sentimental, temporal, content, and language features. Features that describe user identity information, such as the username, are considered to be “static”, as their values do not change over time. However, features such as the number of logins by a user in an IAP are considered “dynamic”, as their values change over time. With such a wide range of features that can be considered, the question becomes whether all these features are significant in discovering malicious bot cyberattacks or whether a particular sub-set of features are significant.

1.3 Research problem

The primary research problem dealt with in this thesis is the intelligent discovery of cyberattacks launched by malicious bots on IAPs, based on anomalous behaviour detection.

1.4 Research questions

The following research questions were formulated to solve the main research problem in Section 1.3:

1. Research Question 1: How to develop or enhance state-of-the-machine learning based cybersecurity solutions to countermeasure cyberattacks launched by bots?
2. Research Question 2: What type of features found on Internet Application Platforms that should be monitored to effectively discover cyberattacks launched by bots?
3. Research Question 3: What cyberattacks are launched by bots currently found on the Internet Application Platforms?
4. Research Question 4: What are the important data quality characteristics to be considered on cybersecurity data sets when designing machine learning based cybersecurity solutions?

1.5 Research scope and methodology

This thesis focused on the intelligent discovery of cyberattacks launched by malicious bots on IAPs. To achieve this, this thesis proposed a methodological approach named CySecML that provides a framework for developing ML-based cybersecurity solutions. The CySecML methodology consists of data preparation and InternetBotDetector model components. The latter model is implemented based on semi-supervised machine learning algorithms. To provide proof-of-concept for this methodology, two different IAPs were considered - firstly, online social networks (OSNs) to discover bot cyberattacks such as social engineering attacks, and secondly, network intrusion detection systems (NIDSs) to discover zero-day network intrusion attacks. For illustrative and benchmarking purposes, the author of this thesis opted to

use publicly available cybersecurity data sets. The configuration work required to integrate this methodology into an existing IAP was not addressed in this current thesis.

The main research problem and the questions raised above were addressed by following the steps discussed below:

1. The first step was to conduct a systematic literature review to understand different types of bots, domains in which they exist, their cyberattacks and the methods for discovering or detecting such bots.
2. The second step was to identify suitable IAPs to study cyberattacks launched by malicious bots, as well as to identify suitable data sets to use and ensure they are of quality.
3. The third step was to review existing feature selection methods, specifically their strengths, limitations and computational costs on cybersecurity data sets.
4. The fourth step involved identifying significant features that are indicative of anomalous bot behaviour.
5. The fifth step was to develop the CySecML methodology and implement prototypes to provide proof-of-concept of this methodology.
6. The final step was to summarise the findings of this research by relating them to the research problem and future work on this topic.

1.6 Structure of the thesis

This thesis consists of the Introduction, Part I, Part II and Conclusion that comprises nine chapters organised as follows:

Introduction

Chapter 1 introduces the thesis with particular focus on cybersecurity threats posed by malicious bots in cyberspace. Moreover, the research problem, research questions and scope of this thesis is outlined.

Part I: Taxonomy of bots and research design

Chapter 2 provides details of the systematic literature review of bots from a cybersecurity perspective. Furthermore, a bot taxonomy that is based on bot type, bot cybercrime, domain and detection or prevention method is proposed.

Chapter 3 discusses related work in relation to the discovery of zero-day network intrusion and social engineering attacks. In addition, research gaps are identified.

Chapter 4 constructs methodology requirement for building machine learning based cybersecurity solutions and develops the CySecML methodology with its components. Furthermore, a pseudo code demonstrating the key steps of the CySecML methodology is presented.

Chapter 5 introduces the cybersecurity data sets used in this thesis to discover bot cyberattacks. Specifically, it outlines how to obtain or create synthetic data for the purposes of training and testing machine learning based cybersecurity solutions.

Chapter 6 presents data quality characteristics that should be considered when developing machine learning based cybersecurity solutions. Furthermore, the cybersecurity data sets used in this thesis are examined in terms of these data quality characteristics.

Chapter 7 provide details of feature selection and machine learning algorithms used in this thesis to intelligently discover cyberattacks launched by bots. In addition, Benford's law is proposed as a feature selection method on cybersecurity data sets.

Part II: Proof-of-concept

Chapter 8 provides proof-of-concept of the CySecML methodology by discovering social engineering and zero-day network intrusion cyberattacks launched by bots.

Conclusion

Chapter 9 concludes this thesis and discusses possible directions for future research.

Figure 1.2 provides a graphical roadmap of this thesis.

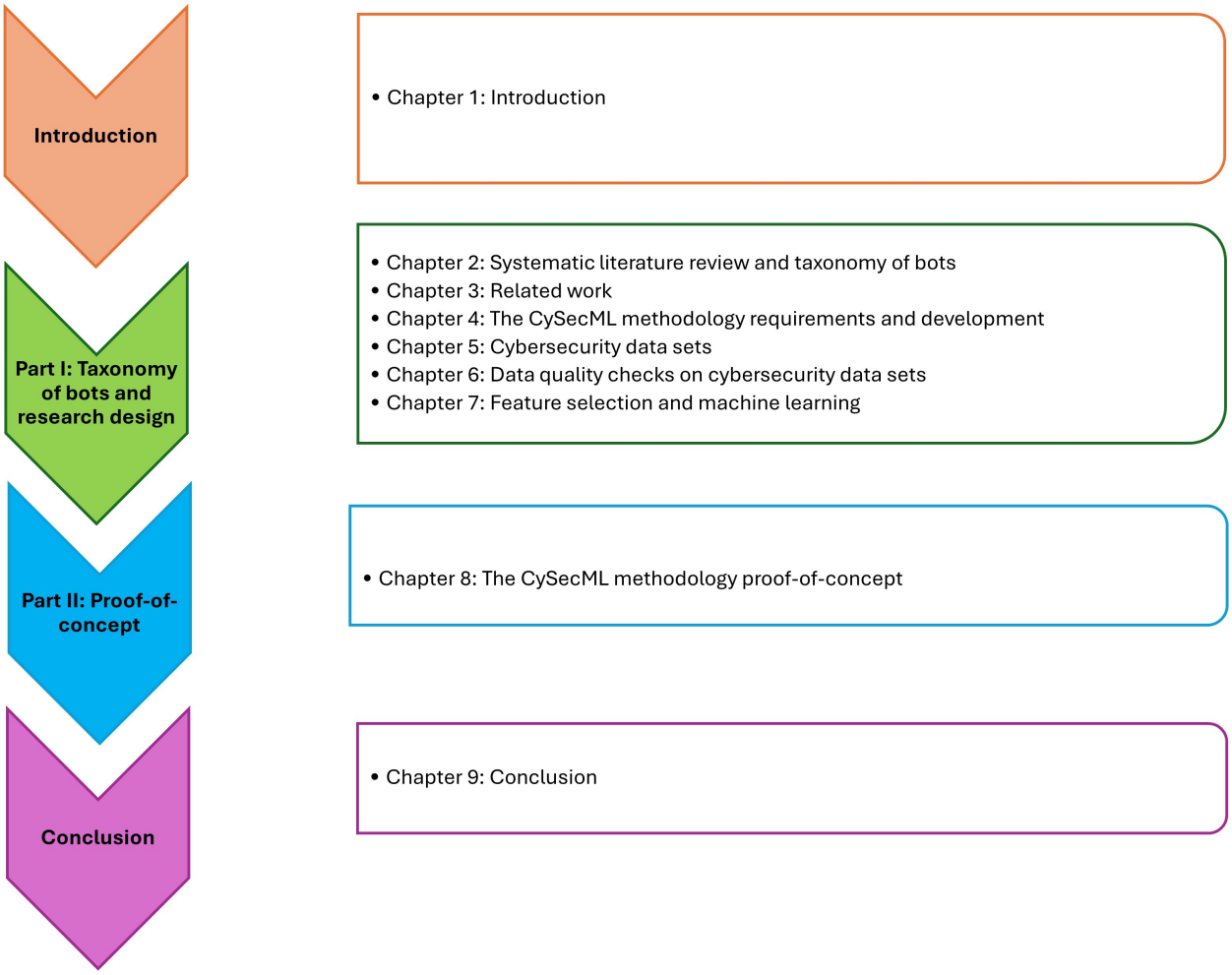


Figure 1.2: Graphical roadmap of this thesis

1.7 Chapter summary

This chapter provides the necessary background information on the thesis and describes the main research problem, which is the intelligent discovery of cyberattacks launched by malicious based on anomalous behaviour detection. Malicious bots impose serious cybersecurity threats as they can launch large-scale attacks much faster than humans can do. Therefore, it is of utmost importance to safeguard cyberspace from the ever-growing number of cyberattacks launched by bots. Malicious bots can launch various attacks in different domains, however, existing literature review studies lack a holistic overview of the cyber threats posed by malicious bots. Therefore, to better understand cybersecurity threats posed by malicious bots, a systematic literature review was conducted and discussed in the next chapter.

Part I

Taxonomy of bots and the CySecML methodology

Chapter 2

Systematic literature review and taxonomy of bots

2.1 Introduction

The state-of-the-art literature on bots lacks a holistic overview of the different types of bots that exist, the cyberattacks they launch, and the domains in which they exist. This chapter aims to fill this void by conducting a systematic literature review (SLR) in accordance with the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) framework. A taxonomy of bots is proposed that aims at providing a high-level overview of different types of bots, their cyber threats, domains and detection or mitigation methods. The steps followed in conducting this SLR are outlined next, followed by the proposed bot taxonomy.

2.2 Systematic literature review (SLR)

A systematic literature review (SLR) is a type of literature study that identifies, collects, and analyses existing research papers to answer specified research question(s) using a rigorous procedure [79, 80]. The main characteristics of an SLR study are planning, e.g., defining research objectives, search processes, e.g., defining search queries, selection processes, e.g., inclusion and exclusion rules, analysis, and reporting [79, 80]. Unlike standard literature reviews, SLRs are reproducible and rigorous [81]. There are many different procedures for conducting an

SLR, all of which aim to achieve the same goal, namely answering a research question rigorously [82, 81]. The author of this thesis chose the PRISMA framework [81] from among many frameworks [79, 80] to systematically review bots from a cybersecurity perspective. The research question addressed in this SLR involved gaining a holistic overview of cybercrimes or cyber threats posed by bots in different domains.

2.2.1 PRISMA framework and bot taxonomy

The majority of existing literature surveys on bots such as those by [83, 65, 84] focus on a specific domain such as OSNs, or cybercrimes such as spam, whereas most cybersecurity researchers would have been interested in gaining a holistic view of cybercrimes launched by different types of bots in different domains, as well as in methods for the detection or mitigation of bots. The author of this thesis is of the opinion that cybersecurity problems do not necessarily exist in silos, but knowledge can be leveraged to address related cybersecurity problems. For example, the understanding of bot cyber threats and discovery methods in websites could benefit the discovery of similar threats in different domains such as OSNs. The search process to address the research question posed in this SLR is defined next.

To better understand cyber threats or cybercrimes posed by bots in cyberspace, the author of this thesis formulated the following search query; (“bot” AND “crime”) OR (“bot” AND “attack”) OR (“bot” AND “detect”) OR (“bot” AND “threat”) OR (“bot” AND “malicious”) OR (“bot” AND “cyber”) OR (“bot” AND “security”) [85]. The “AND” Boolean operator attempted to ensure that only papers that discuss bots and some form of cyber threat or cybercrime were selected. This query was applied to the most popular databases, including Scopus, SpringerLink, IEEE Xplore Digital Library, ACM Digital Library, Web of Science, and DBLP computer science bibliography. The focus of this SLR was limited to studies published between January 2010 and December 2020, and to studies written in English.

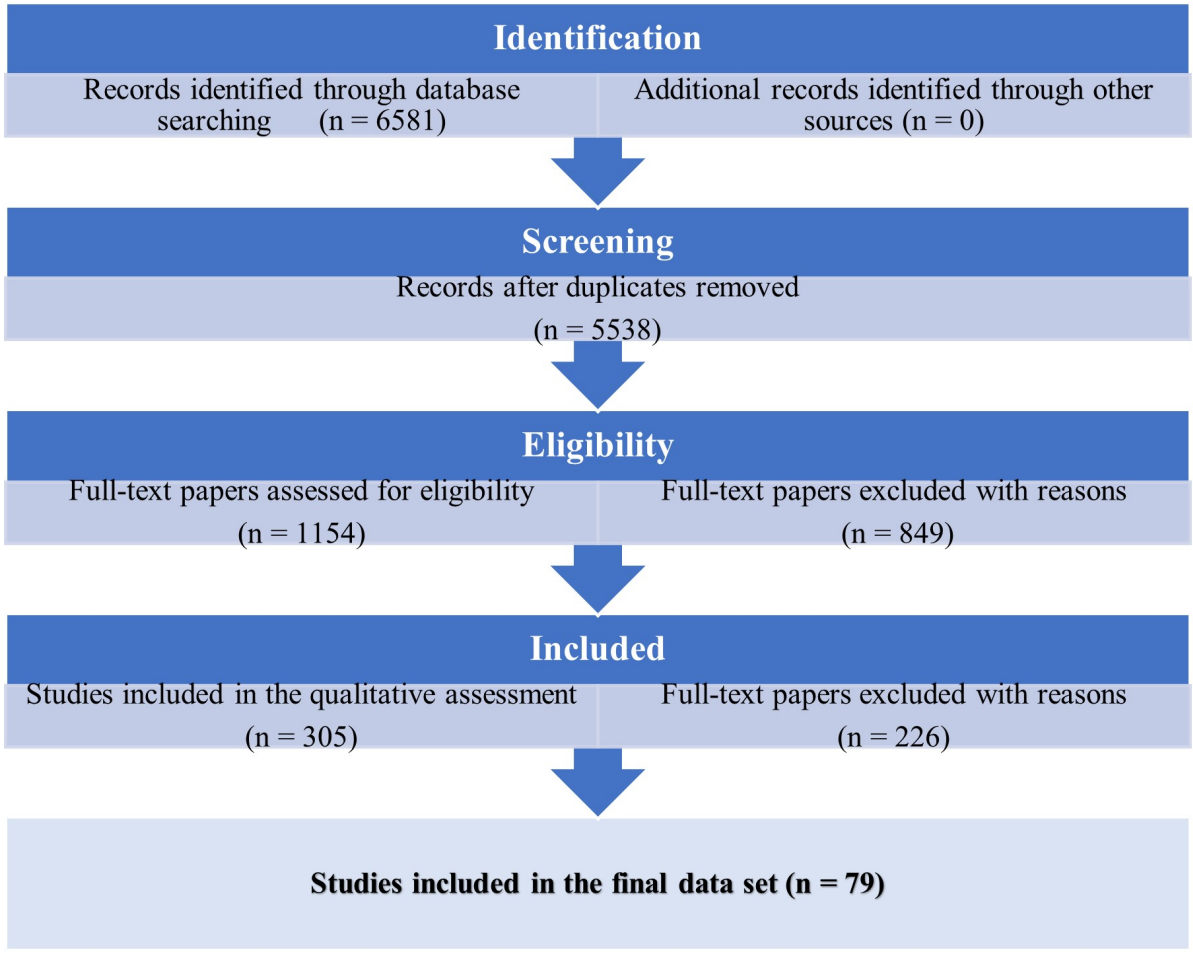


Figure 2.1: Summary of PRISMA framework steps

Figure 2.1 illustrates the steps prescribed by the PRISMA framework. During the identification stage, 6581 papers were extracted from the databases mentioned earlier. At the screening stage, duplicated papers were searched, and 1043 duplicates were identified and checked before being removed. At this stage, 5538 papers remained. To identify eligible studies, the author of this thesis read the abstracts and keywords of each paper. Papers that did not discuss bots and some form of cybersecurity threat or attack were deemed ineligible for this thesis and thus excluded. Altogether 4384 papers were excluded at this stage, and 1154 remained. Further screening was performed in that the paper on each of the remaining studies was read, leading to 849 irrelevant studies being excluded. In the final stage, the remaining studies were categorised into bot type, bot cybercrime, domain, and detection or prevention method. Given that there were relatively many papers in the final stage (305), citations were used as the exclusion criterion. To ensure that this process would not negatively impact the quality of the selected papers, the author of this thesis decided to exclude those with the least number of citations per category, considering publication year of each paper. At this stage, 226 papers were excluded, 79 remained.

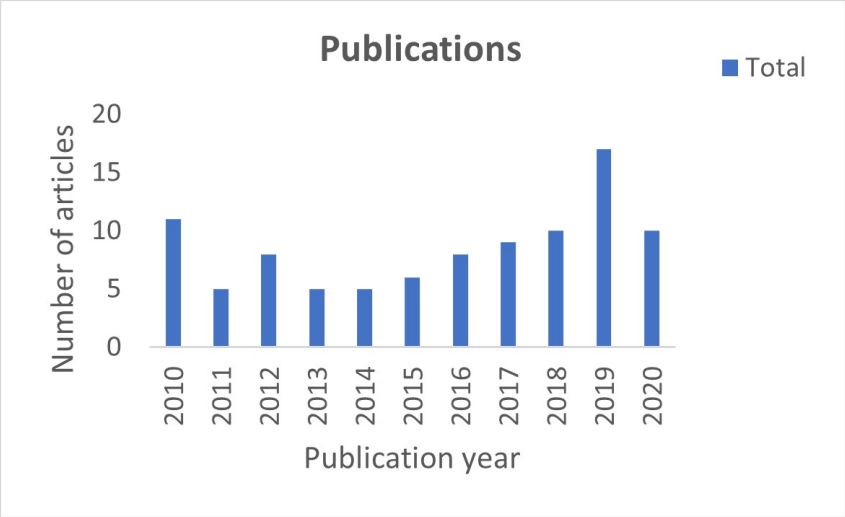


Figure 2.2: The number of articles per publication year, based on the final papers included

Figure 2.2 illustrates the number of papers and publication year of the final papers included, with the highest number of publications being in 2019. The majority of these studies focused on OSN bots because of the general spread of misinformation about the coronavirus outbreak [86].

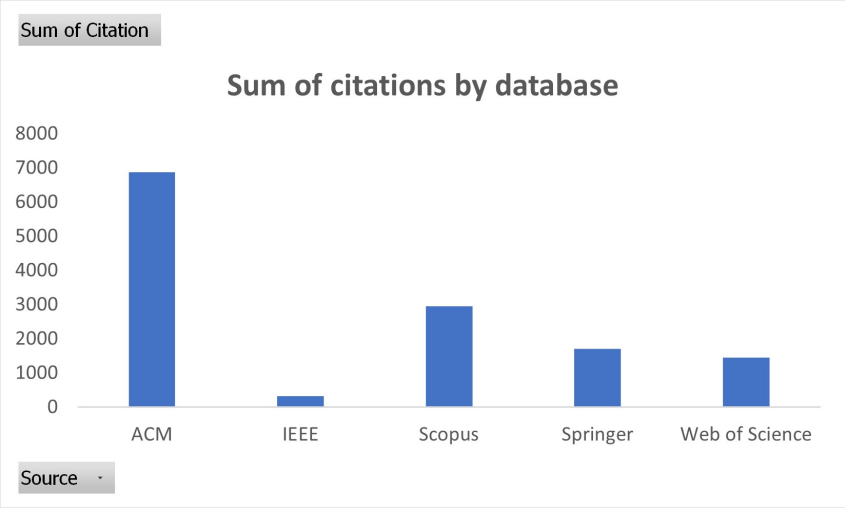


Figure 2.3: The number of citations per database, based on the final papers included

Figure 2.3 indicates that papers from ACM had the greatest number of citations, followed by Scopus, based on the papers included in this thesis. The author of this thesis categorised the 79 papers according to their bot type, bot cybercrime, domain, and detection or prevention methods. Thereafter, a bot taxonomy is proposed based on this categorisation, see Figure 2.4.

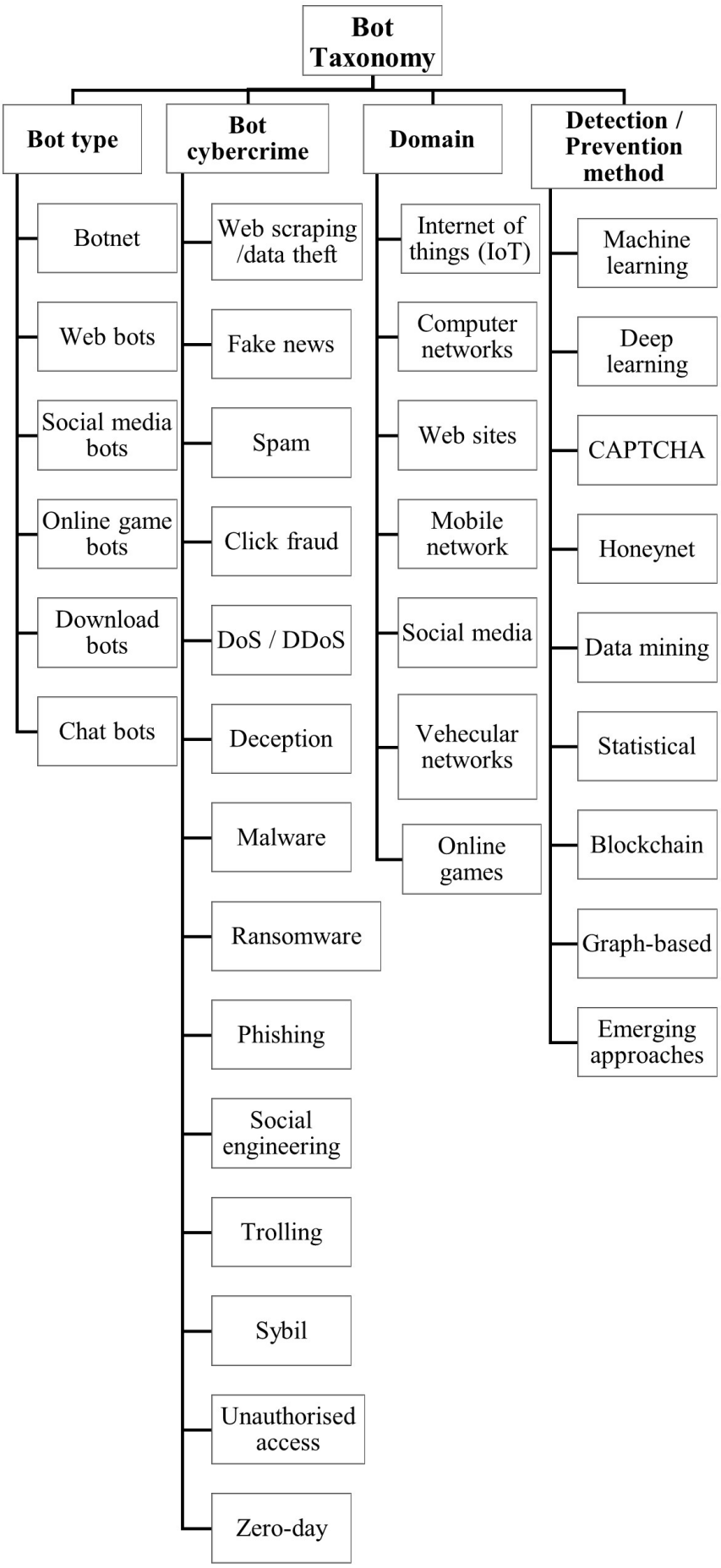


Figure 2.4: Proposed bot taxonomy

The proposed bot taxonomy in Figure 2.4 differs from existing ones because it provides a holistic overview of bots from a cybersecurity perspective. For example, the bot taxonomy proposed by Lebeuf et al. [28] is based on software bots, with a particular focus on the environments in which software bots are deployed. Orabi et al. [84] proposed a bot taxonomy that is based on OSN bots that include Sybil and spam bots. Vitkova [87] et al. proposed a bot taxonomy based on OSN bots with a granular description of bot types such as "Likers" - bots used to like posts, "Reposters" - bots that are used to re-post content among others, and their cyber threats. Yang et al. [88] proposed a bot taxonomy that is based on chat bots with a particular focus on security threats and vulnerabilities such as data breaches. The main limitations of existing bot taxonomies are bot-specific, such as software bots, or domain-specific, such as OSN. The bot types in this proposed taxonomy are described below.

The bot types as per the proposed taxonomy are described below.

- A *botnet* is a collection of compromised bots controlled by a botmaster who uses them to launch cyberattacks such as a distributed denial-of-service (DDoS) attack [89, 90]. Botmasters control their bots either through centralised systems such as command and control (C&C) [91] or decentralised systems such as peer-to-peer (P2P) systems [92, 93]. A centralised system allows a botmaster to communicate directly with each bot in a network, whereas a decentralised system permits multiple communications within a bot network [94].

- *Web bots* also known as Internet bots can be defined as any autonomous computer program that interacts with the Web [95]. There is a multitude of types of Web bots such as online shopping bots, Web search engine bots and Web crawling bots [96]. Most Web bots are benign in nature, e.g., online shopping bots; however, malicious Web bots such as Web crawling bots can be used by cybercriminals to illegally collect confidential user information [69, 97].

- *Social media or OSN bots* are found on OSNs such as 𝕏 and can be used for legitimate purposes, such as sport or news update bots, but also for malicious purposes, such as spreading fake news and spam [98, 99].

- *Online game bots* use technology such as artificial intelligence (AI) to play online games [100, 101]. A cyber threat arises when a person uses a bot to play online games and gain financial rewards under the pretence of him or her playing the game [100].

- *Download bots* are used to download data from the Web. A cyber threat arises when they steal and illegally download massive amounts of data [95].

- *Chat bots* are designed to simulate conversation with humans over the Internet by applying human-computer interaction principles (HCI) [28, 102, 103]. The different types of chat bots range from simple ones such as Apple's Siri, to more sophisticated bots such as vehicular ad hoc network (VANET) bots [103, 104]. Chat bots use natural language processing (NLP) and natural language understanding (NLU) to converse with humans [103, 104]. Although chat bots are mainly used to provide useful services, they are in some cases used for malicious activities such as spreading spam and misinformation [105].

In addition, the bots mentioned above are able to launch various cyberattacks in the form of fake news, misinformation or spam. Figure 2.5 shows the number of cybercrimes committed by bots, such as Web scraping or data theft [96, 97], fake news [34, 106, 7, 51], spam [106, 107, 108, 109, 110, 111], click fraud [112, 113, 114], DoS and DDoS [90, 115, 116, 66, 67], deception [117, 118], malware [119, 120, 121, 122], ransomware [91, 123, 124, 125], phishing [126, 127], social engineering [43, 128], trolling [129], Sybil [130], unauthorised access [131, 132, 133, 134] and zero-day [135].

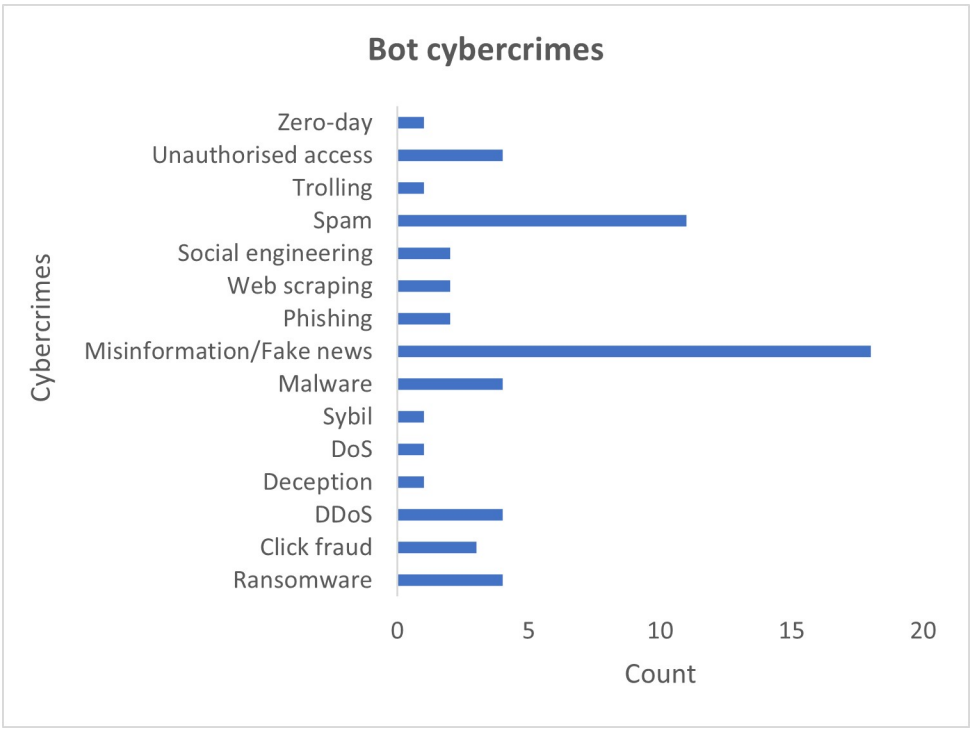


Figure 2.5: Bot cybercrimes

The cybercrimes launched by bots as per the proposed bot taxonomy are described below.

- *Web scraping or data theft* also known as Web extraction or harvesting is a method of extracting raw data from the Web and storing it to a file for future analysis [97, 136]. Various methods exist for extracting data from the Web, ranging from simple "copy-and-paste" to advanced HTTP programming methods. Web scraping is a powerful method for efficiently collecting massive amounts of data on the Web for analysis. Web scraping becomes a cybersecurity threat when data is collected illegally i.e., data theft and used for malicious purposes [137]. A well-known case of Web scraping through bots is the Cambridge Analytica scandal [138], where a data analytics company illegally extracted the profile information of Facebook users.

- *Misinformation or fake news* is an act of spreading falsified information on the Web, OSNs in particular [50]. There is general consensus that the US presidential election campaign in 2016 was influenced by the spreading of misinformation or fake news on OSNs [129, 138].

- *Spam* refers to the act of sending unsolicited information on the Internet [111]. Spam bots are well studied on OSNs [31].

- *Click fraud* occurs on the Web when a user automates the number of clicks on advertisements such that online advertisers pay the user for meaningless clicks [139, 113, 114]. Online advertisers generally use the pay-per-click method to reward users who view their advertisements. Recently, bots were shown to scam online advertisers by automating valueless clicks [114].

- *Dos or DDoS* is an attack to deny legitimate users access to a network by flooding a network server with malicious network connections. In recent years, there has been a significant increase in DoS or DDoS attacks through the use of bots [9, 90, 140].

- *Deception* can be defined as the use of false identity information to mislead others [141, 118]. Malicious bots have been shown to use falsified information on the Web to avoid being discovered [118].

- *Malware* short for malicious software is a general term that describes a range of malicious software programs that compromise users' machine functionality [121].

- *Ransomware* is a malware attack whereby an attacker requires a victim to pay a ransom to de-crypt a malicious program on their machine [124].
- *Phishing* is an act of attempting to illegally obtain users' personal information such as credit card details on the Web by masquerading as a legitimate entity [127].
- *Social engineering* is a manipulation technique that exploits human error to commit security mistakes such as clicking on a malicious URL [142, 143]. In the case of OSNs, bots often launch social engineering attacks by using fake social engagement [18, 45], e.g., purchasing fake followers on black markets to make their accounts appear popular so they can trick humans once they have earned their trust [128, 144].
- *Trolling* is an act of provoking or manipulating others on the Web by posting inflammatory messages that might cause conflict [129, 145]. Users who display such behaviours are referred to as "trolls" [146].
- *Sybil* is a type of attack in which an attacker uses multiple fake identities to dominate a large portion of the network [147, 148].
- *Unauthorised access* is an attempt to access websites or systems illegally [134]. It is common for websites to implement CAPTCHA methods to restrict access to their websites to humans only [149].
- *Zero-day* is a generic term that describes a new type of attack not witnessed before [59, 150].

Next, the domains in which these bot cybercrimes are launched were analysed as per the proposed bot taxonomy. Figure 2.6 shows that most bots are found on OSNs and computer networks.

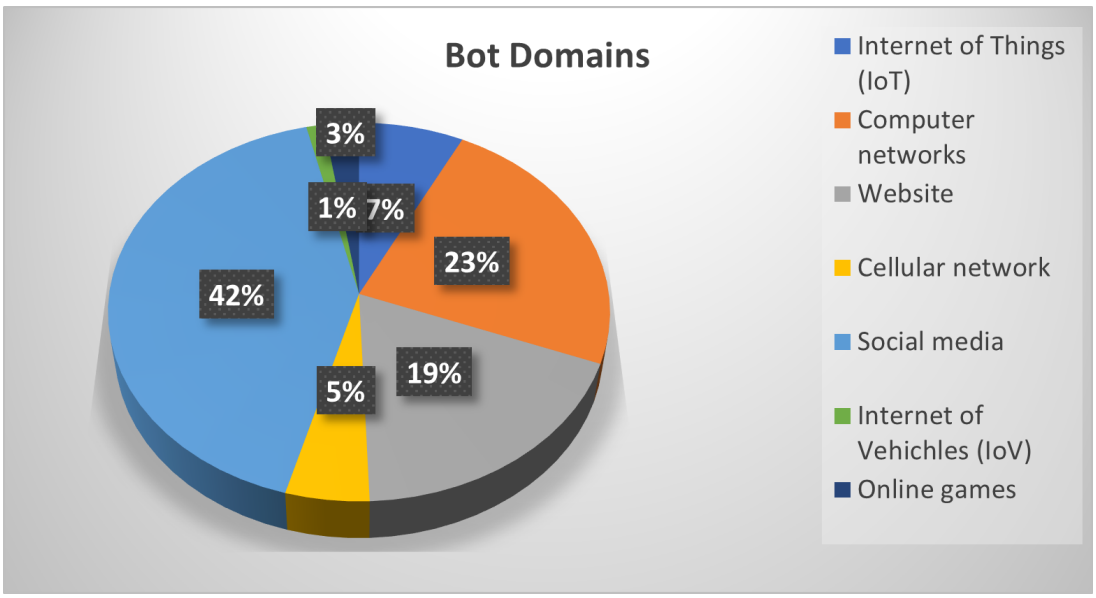


Figure 2.6: Bot domains

Finally, the methods for detecting or preventing these bot cybercrimes were categorised under machine learning [96, 151], deep learning [70, 114, 152], CAPTCHA [134, 153], Honeynet [108, 154], data mining [121], statistical [155, 130, 99], blockchain [156], graph-based [157], and emerging approaches such as ensemble methods [135] as per the proposed bot taxonomy.

The proposed bot taxonomy in Figure 2.4 aims to address one of the research questions of this thesis which is to provide a holistic overview of current cyberattacks launched by bots. Among the various bot cyberattacks identified in the proposed bot taxonomy, the author of this thesis focused on zero-day and social engineering cyberattacks, as existing research in this regard was found to be limited.

In the next chapter, a detailed analysis of related studies on zero-day network intrusion and social engineering attacks is presented.

2.3 Chapter summary

This chapter presented a systematic literature review and bot taxonomy from a cybersecurity perspective consisting of bot type, bot cybercrime, domain, and detection or prevention methods. The proposed bot taxonomy forms part of the building blocks to address the problem of intelligently discovering cyberattacks launched by bots on IAPs. Among the various types of

cyberattacks launched by bots, this chapter revealed a shortage of studies focusing on social engineering and zero-day network intrusion attacks launched by bots in OSNs and NIDSs, respectively. Henceforth, this thesis limits its focus to these bot cyberattack types. This systematic literature review and bot taxonomy do not provide insights into the strengths and limitations of existing cybersecurity solutions in discovering these bot cyberattacks; therefore, the next chapter critically evaluates related studies and identifies research gaps.

Chapter 3

Related work

The previous chapter identified studies related to the discovery of zero-day network intrusion and social engineering attacks in NIDSs and OSNs respectively. These studies are now critically evaluated to identify research gaps. The purpose of this discussion is to assess cybersecurity data sets, significant features, and the performance of methods used to discover zero-day network intrusion and social engineering attacks. This analysis was used to benchmark the InternetBotDetector model in the upcoming chapters. Related studies on zero-day network intrusion attacks in NIDSs are discussed next, followed by social engineering attacks in OSNs and the identified research gaps.

3.1 Bot cyberattacks in NIDS

3.1.1 Introduction

A firewall is a security solution that controls incoming and outgoing traffic of computers from a network, based on predefined security rules such as the analysis of IP addresses [158, 159]. In contrast, NIDSs are security solutions that are located at specific points within a wireless network to monitor data packets transmitted across the network in order to identify suspicious or malicious activities within a network [160, 161]. The main difference between a firewall and a network intrusion detection system is that NIDSs continuously monitor network traffic, whereas a firewall grants or denies computer access to a network only at the point of request [159, 162]. Both firewalls and NIDSs form part of the cybersecurity framework, which refers

to the protection of computers and electronic data from threats to the CIA triad of the systems [162, 163]. As stated in the introduction chapter of this thesis, this thesis focuses on NIDSs. NIDSs are used to discover cyber network intrusion traffic by analysing data extracted from network devices [67, 164]. Research on the development of NIDSs dates back to the 1980s [165] when Anderson [166] developed an NIDS that monitored users' normal behaviour. Thus, user behaviour that deviated from the expected normal behaviour was deemed an anomaly and therefore could be malicious. Since then, many research efforts have been made to discover or detect network intrusion cyberattacks, and a few key studies were sampled in this thesis.

3.1.2 Zero-day network intrusion attacks - related studies

Vahdani et al. [167] proposed a hybrid unsupervised approach for discovering zero-day network intrusion attacks. This hybrid approach consists of self-adaptable and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) techniques. The self-adaptable method discovers anomalies, i.e., network attacks by training a model to learn normal network traffic well, so that network traffic that deviates from this normal behaviour based on a certain threshold is deemed to be an anomaly. The advantage of this approach is that the threshold is self-adaptable to the current status of network traffic, as a predefined fixed threshold may result in high false-positive or negative rates as network traffic data continuously evolves. Once the self-adaptable method identifies attacks, the DBSCAN method attempts to cluster such attacks by using features such as the IP addresses with the aim of discovering a botnet or botmaster. The limitation of their approach is that cybercriminals often hide their true IP addresses by using a proxy server to avoid being discovered. Vahdani et al. [167] used the DAPRA and ISCX network intrusion data sets for illustrative purposes. They achieved an accuracy of 98% and a false-positive rate of 3.61% for discovering zero-day network intrusion attacks.

Duong et al. [168] proposed a semi-supervised learning approach based on a modified Mahalanobis distance principal component analysis (M-PCA) to discover zero-day network intrusion attacks. In this setup, the authors implemented the PCA and K-means clustering method in the training phase to build a normal or expected network traffic profile by removing outliers, i.e., malicious network traffic. Thereafter, M-PCA was used in the testing phase to discover zero-day network intrusion attacks. Mahalanobis distance[1] is a measure of distance d between

[1] $d(p,\mu) = \sqrt{(p-\mu)^T S^{-1}(p-\mu)}$ where S^{-1} is the sample covariance matrix [168].

the observed point p and sample mean μ. The computational cost of S^{-1} can be expensive for high-dimensional data sets, which is a disadvantage. Duong et al. [168] used the NSL-KDD data set to evaluate the proposed model. The M-PCA algorithm achieves an accuracy of 91%.

Pallaprolu et al. [169] presented a hybrid approach based on semantic link networks (SLN) and dynamic semantic graphs (DSG) to discover zero-day network attacks. By using a network intrusion data set that contains both benign and malicious network traffic, the algorithm creates an SLN where each node represents malicious network traffic, and the edges represent the semantic relationship between malicious network traffic. This phase was completed as part of the training phase of the model. To mimic a zero-day attack scenario, the authors excluded some malicious network traffic in the training phase and only included them in the testing phase. The well-known limitation of SLN is the calculation of weights used to connect nodes [169] - particularly for zero-day attacks. Pallaprolu et al. [169] used the KDD CUP'99 data set, which contains 23 network intrusion attacks including DoS attacks to evaluate their approach. The minimum redundancy maximum relevance (mRMR) feature selection method was implemented to identify significant features and the mRMR method identified 25 significant features from 42 features. The authors achieved an accuracy of 98% in discovering zero-day network attacks.

Zhu et al. [170] proposed a hybrid semi-supervised learning strategy based on the k-nearest neighbour (kNN) and decision tree to discover zero-day intrusion attacks. In this semi-supervised strategy, the ensemble classifier is trained using the kNN and decision tree algorithms on the NSL-KDD network intrusion data set. Zhu et al. [170] then simulated malicious network traffic without labels to create a test data set. If the majority of the classifiers deemed a particular network traffic to be malicious, e.g., a zero-day attack, then the ensemble classifier would declare with high confidence that network traffic was malicious. All data points declared malicious with high confidence could then be added to the training data set to refine the learning phase of the classifier. This ensemble classifier achieved an accuracy of 80% in the discovery of zero-day network intrusion attacks. Their approach was tested on only one data set; hence, their findings cannot be generalised.

Aygun et al. [171] proposed a hybrid anomaly-based approach that combines autoenconder (AE) and denoising autoenconder (DAE) to discover zero-day network intrusion attacks. In this approach, AE is used in the training phase and DAE is used in the testing phase of a model to

overcome the known overfitting problems of AEs. The authors used the NSL-KDD network intrusion data set to achieve an accuracy of 88%. Their findings cannot be generalised as the authors used only one data set.

Chiba et al. [160] proposed an anomaly-based NIDS to discover zero-day network intrusion cyber threats. Their approach was based on the back-propagation neural network (BPNN) algorithm. They evaluated the BPNN algorithm using the KDD CUP'99 network intrusion data set, which contains 41 total features of symbolic, numeric and Boolean data types. Chiba et al. [160] implemented the Kolmogorov-Smirnov correlation-based (KSC) filter and information gain (IG) feature selection methods. The KSC method identified 12 significant features of which 10 were numeric, and the IG method identified 17 significant features of which 12 were numeric. Moreover, there were five common numeric features between the two feature selection methods. The authors also selected all 34 numeric features as the input variables in the BPNN algorithm. Throughout their experiments, the BPNN model that used 34 numerical features outperformed the other methods with an accuracy rate of over 98%. This demonstrates that numeric features such as source destination bytes are significant for the discovery of anomalous behaviour, as opposed to non-numeric features such as protocol, which display similar behaviour for normal and anomalous network traffic.

Abri et al. [72] investigated the performance of 34 machine learning (ML) and deep learning (DL) algorithms in discovering zero-day attacks. They evaluated these models using the Meraz'18 data set, which contains benign and malicious records described by 55 features. Abri et al. [72] did not implement any feature selection; instead, they manually removed three features with static data. They found that the random forest ML algorithm produced the best results with an accuracy of 99.51%, followed by the decision tree model with an accuracy of 99.24%. The Gaussian naïve Bayes ML algorithm was the least performing model with an accuracy of 46.31%. DL algorithms such as the multi-layer perceptron (MLP) performed well, with an overall accuracy of over 98%. Abri et al. [72] demonstrated that both ML and DL algorithms can be effective in discovering zero-day attacks. The authors did not specify the significant features that contributed the most to these results.

Taher et al. [172] proposed a supervised ML approach based on an artificial neural network (ANN) to discover zero-day network intrusion attacks. They used the NSL-KDD network intrusion data set with its four main cyber intrusions to evaluate the proposed model: denial-of-

service (DoS); remote to local (R2L); user to root (U2; and probe). For each main network intrusion attack there are associated sub-classes of network attacks described by 43 features. The ANN model achieved a detection rate of 94%. The limitation of using a supervised ML approach for discovering zero-day attacks is that signatures of zero-day attacks are unknown; therefore, it is practically impossible to train an ML model on all possible types of zero-day attacks.

Blaise et al. [173] proposed a split-and-merge algorithm and used an unsupervised port-based learning approach for discovering zero-day network intrusion attacks. The split-and-merge algorithm consists of two phases. The first phase is the splitting of a network segment into sub-regions using each feature and port based on a modified Z-score measure. If the Z-score measure exceeds a certain threshold, the sub-networks are further divided until no further splitting is possible. The second phase merges all network segments that have a common property [173]. Blaise et al. [173] used the MAWI and UCSD network intrusion data sets to evaluate the proposed model. The split-and-merge algorithms achieved positive rates of 79% and 86% for the MAWI and UCSD data sets respectively. Unsupervised approaches can suffer from high false alarm rates if the model fails to split benign and malicious network traffic [174].

Zavrak et al. [175] implemented autoencoder (AE), variational autoencoder (VAE), and one-class support vector machine (OCSVM) models to discover zero-day network intrusion attacks. The AE and VAE models are based on deep learning (DL) methods. The authors adopted a semi-supervised learning strategy to discover zero-day attacks. In this strategy, the original network intrusion data set was divided into training and testing data sets. The training data set consisted of only normal labelled network traffic, whereas the testing data set consisted of unlabelled data sets of normal and malicious network traffic. Zero-day attacks were created by ensuring that the data points used in the testing phase were different from those used in the training phase [175]. Although the unlabelled data points in the testing phase were known, this strategy was decided upon by Zavrak et al. [175] in order to evaluate the proposed models.

Zavrak et al. [175] used the CICIDS2017 network intrusion data sets that contained a variety of malicious network traffic such as DoS hulk, Port scan, DDoS, DoS goldeneye, FTP-bruteforce, SSH-bruteforce, DoS slowloris, DoS slowhttptest, botnet, heartbleed, infiltration, Web attack-brute force and Web attack-XSS. The benign and malicious network traffic in this data set is described by 80 features in flow-based and packet-based formats. The VAE achieved an area

under the ROC curve (AUC) of 76%, AE of 74%, and OCSVM of 66% for discovering zero-day attacks. However, these models achieved higher AUC results for specific attacks. Specifically, the OCSVM achieved 99% on heartbleed attacks, VAE achieved 90% on infiltration attacks, and AE achieved 83% on DoS hulk attacks. Interestingly, the AUC results for all models were higher for these specific attacks and lower for the zero-day attacks. These results demonstrate the need for further research to discover zero-day attacks intelligently.

Hindy et al. [176] implemented AE and OCSVM models to discover zero-day network intrusion attacks, similar to the work done by Zavrak et al. [175]. Hindy et al. [176] used the CICIDS2017 and NSL-KDD network intrusion data sets to evaluate their proposed models. They used an unspecified method to remove highly correlated, i.e., over 0.9 threshold features from normal network traffic. Thereafter, the proposed models were trained solely based on normal network traffic. The AE approach had an accuracy of up to 98% on the CICIDS2017 data set and 93% on the NSL-KDD data set. On the other hand, the OCSVM achieved a low accuracy score of up to 57% on the CICIDS2017 data set and 88% on the NSL-KDD data set.

Pu et al. [177] presented a hybrid unsupervised anomaly-based clustering detection method for discovering zero-day network intrusion cyber threats. The hybrid SSC-OCVM model is based on a combination of sub-space clustering (SSC) and OCSVM. The SSC model is responsible for clustering unlabelled data into specific sub-spaces according to a similarity measure, whereas the OCSVM model is responsible for training the model using only benign network traffic classes. Pu et al. [177] evaluated their proposed approach using the NSL-KDD network intrusion data set. They implemented an F-test feature selection method and achieved a detection rate of up to 99% for zero-day attacks. However, it is not clear from their paper which features contributed significantly to these results.

Zoppi et al. [178] evaluated 17 different unsupervised ML algorithms, including DBSCAN and isolation forest (iForest), to discover zero-day network intrusion attacks. They used the SDN20 network intrusion data set to evaluate the models. This data set contains five malicious network traffic types: DoS, DDoS, probe, brute-force, and exploits. Benign and malicious network traffic was described by 85 features. Zoppi et al. [178] used the rapid evaluation of anomaly detectors (RELOAD) [179] tool to identify significant features, including backward header length and packet average size. The iForest algorithm achieved the best results with a Matthews correlation coefficient (MCC) of 89%. The unsupervised ML challenges have been

discussed in this section.

Abdalgawad et al. [180] implemented various DL models that included adversarial autoencoders (AAE) and bidirectional generative adversarial networks (BiGAN) for the discovery of network intrusion attacks, including zero-day attacks. They used the IoT-23 network intrusion data set to evaluate their proposed models. The IoT-23 data set contains 15 network intrusion attacks, such as DDoS and botnets, which are described using 19 features. After Abdalgawad et al. [180] performed data cleaning and feature selection, only eight features were deemed significant. Moreover, they implemented the synthetic minority oversampling technique (SMOTE) because this data set was highly imbalanced [180]. The BiGAN method achieved an F_1 score greater than 85% for the discovery of zero-day attacks. Data balancing techniques such as SMOTE run a risk of discarding important information [35, 181], therefore, feature selection method that can deal with imbalanced data sets should be sought.

Arshadi et al. [182, 183] proposed the use of Benford's law to analyse Internet network traffic. They demonstrated that normal network traffic tends to obey the first significant leading digit (FSLD) property of Benford's law, whereas network attacks such as port scan which affect the inter-arrival times of TCP network flows violate the FSLD. Arshadi et al. [182, 183] used the Measurement and Analysis of Wide Internet (MAWI) data set in their experiments. Furthermore, they established that the FSLD of Benford's law was consistent with the Weibull distribution for a given shape and scale parameters. Further research is required to validate the use of Benford's law on cybersecurity data sets.

Iorliam [184, 185] demonstrated that the difference between two consecutive TCP network traffic flows, called "flow-size difference" can be used to discover malicious network traffic. Specifically, the authors of [184] demonstrated that the flow size difference in normal network traffic tends to obey the FSLD of Benford's law, whereas malicious network traffic affecting TCP flows violates it. Similar findings about the TCP flow-size differences were observed in [186, 187].

The above studies are summarised based on data sets, performance types, discovery type and methods used to discover zero-day network intrusion. The summary in Table 3.1 is used as a benchmark against InternetBotDetector in the upcoming chapters.

Authors	**Data set**	**Performance**	**Discovery type**	**Method**
Pu et al. [177]	NSL-KDD	Detection rate - 99%	Anomaly-based	Sub-space clustering & OCSVM
Chiba et al. [160]	KDD CUP'99	F_1 score - 99%	Anomaly-based	Backpropagation neural network
Zoppi et al. [178]	SDN20	MCC score - 89%	Anomaly-based	iForest
Taher et al. [172]	NSL-KDD	Accuracy - 94%	Signature-based	Artificial neural network
Abri et al. [72]	Meraz'18	Accuracy - 99%	Signature-based	Random forest
Vahdani et al. [167]	DAPRA & ISCX	Precision - 98%	Anomaly-based learning	DBSCAN clustering
Hindy et al. [176]	CICIDS2017 & NSL-KDD	Accuracy - 99%	Deep learning	Artificial neural network
Abdalgawad et al. [180]	IoT-23	F_1 score - 85%	Deep learning	AAE and BiGAN
Blaise et al. [173]	MAWI & UCSD network	TP rate - 86%	Anomaly-based learning	Split and merge
Pallaprolu et al. [169]	KDD CUP'99	Accuracy - 98%	Signature-based	SLN and DSG
Duong et al. [168]	NSL-KDD	Recall - 89%	Hybrid-based	Mahalanobis distance PCA
Zhu et al. [170]	NSL-KDD	Accuracy - 80%	Hybrid-based	Decision tree and KNN
Zavrak et al. [175]	CICIDS2017	AUC - 76%	Anomaly-based	Variational autoencoder
Aygun et al. [171]	NSL-KDD	Accuracy - 88%	Anomaly-based	Autoencoder and denoising autoencoder

Table 3.1: A summary of existing methods for discovering zero-day attacks

The definitions of performance measures are found in the glossary chapter. The results in Table 3.1 show that anomaly-based approaches are those most commonly used for discovering zero-day network intrusion attacks. This is primarily attributed to the unsupervised ML capabilities of learning with unlabelled data [177, 160]. Of particular interest in this thesis are the results by Duong et al. [168] and Zhu et al. [170] who proposed a semi-supervised ML approach for discovering zero-day network intrusion attacks. See Ahmad et al. [188] for further details on methods used to discover zero-day network intrusion attacks.

The next section provides a comprehensive analysis of cyberattacks launched by bots in OSNs. Similar to the previous section, this analysis was used to benchmark the InternetBotDetector model in discovering cyberattacks such as social engineering attacks in OSNs.

3.2 Bot cyberattacks in OSN

3.2.1 Introduction

In general, OSNs such as 𝕏 implement methods such as multi-factor authentication and CAPTCHA methods to block non-human users from accessing the platform [189]. The owner of a bot account (botmaster) assists in registering its account, including providing profile details [21]. Once an account is registered, it is permitted to virtually interact with other users of the platform by posting content and following other accounts amongst many other activities [190, 191, 87]. A botmaster can automate the activities of its bot account, such as sending unsolicited (spam) messages to particular users [111, 192].

OSN platforms are open in nature, in a sense that anyone can create an account without intensive verification processes such as providing identity identification documents [21]. This "openness" and the ease of registering new accounts have made OSNs targeted by cybercriminals who use them to launch cyberattacks or promote a particular agenda, as in the case of the Venezuela elections that were influenced by malicious bots [52]. cyberattacks launched by bots in OSNs are a long-standing problem [45] that has a serious impact on individuals and organisations that use OSNs. To date, malicious bots pose serious cybersecurity threats in OSNs, given the various cybercrimes they launch, which impact the lives of people [38]. These cybercrimes

are highlighted in the next section. Not all bots found in OSNs are used for malicious purposes such as spreading fake news to influence public opinion [86, 193, 194] or Web scraping [195, 196]. Some bots are used for benign purposes, such as weather updates [43, 34, 197].

The below section discusses the common behavioural differences and similarities between benign and malicious bots.

3.2.1.1 Benign versus malicious bots

Both malicious and benign bots can operate under automation using an application programming interface (API) in OSNs [4]. However, their behavioural activities such as posting patterns (features and intentions) may differ [4, 30]. Benign bots, such as news update bots, share breaking news, whereas malicious bots, such as spamming bots, spread spam [4]. In practice, it is difficult to determine upon registration of an account whether a bot has malicious or benign intentions [34]. The common approach to differentiate between the two is to analyse their behaviours and activities [4, 34], and this is the approach adopted in this thesis . Table 3.2 highlights some of the similarities and differences between the known behaviours of benign and malicious bots.

Benign bots	**Malicious bots**
Perform automated tasks, such as posting tweets.	Perform automated tasks, such as posting tweets.
Perform benign activities, such as posting weather updates.	Perform malicious activities, such as spamming.
Contribute to network traffic.	Contribute to network traffic.
Often declares on the profile that it is a bot account.	Often hides that it is a bot account (to mislead the public).
Generally, is controlled by a single user.	Generally, is controlled by a botmaster and part of a botnet network.
Often, shows a single IP address within an Internet connection session [69].	Often shows multiple IP addresses within an Internet connection session [69].
Generally, has a low rate of failed login attempts [198].	Generally, has a high rate of failed login attempts [198].

Table 3.2: Behavioural differences and similarities between benign and malicious bots

The similarities and differences between benign and malicious bots, as indicated in Table 3.2, can cause these bots to be misclassified. As part of the research problem at hand, the following additional research questions were formulated to inform this thesis.

- Are the features used to differentiate successfully between malicious bots and humans equally useful for differentiating between benign and malicious bots?
- What features found in the meta data of OSNs indicate anomalous behaviour between benign and malicious bots?
- Can semi-supervised machine learning algorithms be used to classify malicious and benign bots, given a limited labelled data set of such bots?

The next section highlights a sample of the key related studies.

3.2.2 Malicious bot attacks - related studies

In a survey of bots found in OSNs, Latah [65] focused on papers published between 2006 and 2019. Latah [65] characterised bot attacks in three stages: initiation, listening, and execution. Latah [65] subsequently categorised bot attacks into 11 categories which included Sybils, crawling, profile cloning, deception, phishing, and spamming. The attacks were categorised at different stages from inception to execution. For instance, the creation of fake profiles and profile cloning falls in the initiation stage, deception falls in the listening stage, and phishing falls in the execution stage. Moreover, Latah [65] proposed a bot detection taxonomy based on graph, i.e., random-walk approaches and ML i.e, supervised, unsupervised, and semi-supervised approaches, as well as on emerging approaches that included a combination of graph-based and ML-based approaches. Their survey was limited to OSN bots.

Orabi et al. [84] conducted an SLR study of OSN bots. They focused on papers published between 2010 and 2019 and proposed a bot detection taxonomy based on graph, ML, crowdsourcing, and anomaly detection techniques. Under ML-based approaches, supervised and unsupervised algorithms were further categorised as behaviour-based and content-based approaches. Behaviour-based approaches use account behavioural features, such as accounts actions and interactions, whereas content-based approaches examine textual features such as tweet content to classify bot and human accounts. Orabi et al. [84] indicated that supervised ML approaches such as random forest (RF) are popular for discovering social media spam, Sybils, and cyborg bots. Cyborg bot accounts include both half-human and half-bot accounts [199, 200, 145]. Similary, their study was limited to OSN bots.

Chu et al. [199, 200] proposed a supervised random forest ML algorithm for classifying 𝕏 users in three distinct categories: humans, bots, and cyborgs (half human and bot). Although their proposed model produced mediocre results in classifying bots and cyborgs, it successfully distinguished between human accounts and bot accounts. These results highlight the difficulty in classifying bot-type accounts such as benign or malicious bot types.

Ferrara et al. [201, 202] proposed using a supervised RF ML algorithm based on over one thousand features to develop a tool for classifying 𝕏 accounts as either human or bot. At present, this tool is called botometer, previously known as *BotOrNot*. The botometer tool extracts features from the meta data of account content, friends, sentiment, timing, and network.

The botometer produces an output score in the range $[0, 1]$. A lower score, i.e., closer to zero indicates a high probability that the account in question is controlled by a human, whereas a high score, i.e., closer to one indicates a high probability that the account in question is a bot. Moreover, the bots in the botometer were assumed to be spam. The botometer tool is freely available online [201, 202] and is primarily used by practitioners and academics. Van der Walt et al. [203] classified fake identities created by humans versus bots using supervised machine learning approaches that include SVM, Adaboost and random forest. The random forest produced best results with an accuracy of 87% and F_1 score of 50%.

Cresci et al. [204] studied various types of malicious bots on 𝕏, such as traditional spam bots, fake followers, fake advertisements, and retweet spams. Moreover, they evaluated the accuracy of human judgement, 𝕏 itself, and various techniques including botometer in classifying malicious 𝕏 bots. Their results demonstrated that the classification of malicious 𝕏 bots is still a challenge, as the accuracy based on human judgement was 24% for social spam bots and botometer, with an F_1 score of 29%. Finally, Cresci et al. [204] found that a significant proportion of spam bots were still active at the time of their investigation.

Khaled et al. [205] used a hybrid supervised machine learning approach to investigate fake i.e., Sybil 𝕏 accounts. Their hybrid algorithm was based on support vector machines and neural network models called SVM-NN, and they implemented PCA and correlation feature selection methods, among others. In this hybrid model, the SVM was used for training and NN for testing purposes. At best, the SVM-NN achieved an accuracy of 98% based on the significant features identified by the correlation feature selection method. The PCA method is computationally expensive for big data platforms [206, 207].

Wang [208] evaluated various supervised ML algorithms, including SVM, neural networks (NN), naïve Bayes and decision tree methods, for classifying 𝕏 accounts as spam or non-spam. Wang [208] collected the data set using 𝕏's API and a Web crawler to extract user meta data such as the most recent tweets. Wang [208] proposed the use of three content-based features, i.e., the number of mentions or replies; duplicated tweets and HTTP links and three graph-based features, i.e., the number of friends, followers and follower ratio[2]. Accounts with a small number of followers compared with the accounts they follow, are generally considered spam

[2] $follower\ ratio = \frac{accounts\ you\ are\ following}{accounts\ that\ follow\ you}$

[6, 209, 155]. Content-based features include the number of duplicated contents, i.e. retweets, replies or mentions, and HTTP links. Similarly, spam bot behaviours include accounts that more often retweet content than share original content [155, 210, 146]. The decision tree, NN, SVM and naïve Bayes methods achieved F_1 scores of 44%, 59%, 40%, and 92% respectively for classifying spam bot and non-spam 𝕏 accounts. These mediocre results demonstrate the need for further research.

Freeman et al. [211] evaluated SVM, RF and logistic regression (LR), and used a supervised learning algorithm to classify clusters of OSNs as fake or legitimate. The accounts were clustered using pattern-encoding algorithms. Freeman et al. [211] used the LinkedIn data set, which contains information about the registration date of the account and the registration IP address. The RF algorithm achieved the best results with an AUC score of 98%. Features that included account descriptions and email addresses were identified as most significant in this algorithm. Gilani et al. [212, 213] collected a large-scale 𝕏 data set that comprised human and bot accounts. Gilani et al. [212, 213] categorised this data set into four groups, using the follower_count feature as a measure of popularity, and observed that well-known accounts with a high follower_count have fewer bots, compared to normal accounts with fewer follower_count measures. Moreover, Gilani et al. [212, 213] used the botometer tool to classify human and bot accounts. The botometer achieved an accuracy of about 51%. Similarly, these mediocre results demonstrate the need for further research.

Oentaryo et al. [4] implemented naïve Bayes, SVM, random forest, and logistic regression using supervised machine learning algorithms to classify humans, benign bots, i.e., broadcast and consumption bots, and malicious bots i.e., spam-promotion and spam-trick bots using a 𝕏 data set. Oentaryo et al. [4] maintained that their study was the first to classify OSN bots as either benign or malicious. The logistic regression model performed best and obtained an F_1 score of 74% for classifying OSN bots. Using Shannon entropy, features such as tweets, retweets, hashtags, mentions, and URLs were identified as significant features for classifying benign and malicious bots. Akyon et al. [128] investigated fake social engagements [128, 137, 214] such as the dissemination of fake news, content, and fake profiles caused by bots. Oentaryo et al. [4] implemented various supervised ML algorithms that include SVM, naïve Bayes, neural networks and logistic regression. The neural network algorithm performed best with an F_1 score of 89%.

Rodríguez-Ruiz [215] proposed using one-class approaches to discover malicious bots on 𝕏. One-class classification approaches train the classifier using only one class, e.g., legitimate users, and test a new data point if it belongs to this class. Rodríguez-Ruiz [215] evaluated the proposed models using the 𝕏 data set provided by Cresci et al. [204] and achieved an AUC of 89%. Dorri et al. [216] developed a tool called SocialBotHunter to classify human and spam bot accounts. SocialBotHunter is based on OCSVM, which adopts a semi-supervised learning approach. The three key functionalities of SocialBotHunter are feature extraction, anomaly prediction score, and botnet detection. Dorri et al. [216] used the 1KS-10KN 𝕏 data set to evaluate their approach, whereby features such as the number of repeated tweets, account age (in days), and average number of spam words in tweets were used as input variables. The average number of spam words can be computationally expensive because bots have been shown to use dynamic words that do not belong to a list of common spam words [111, 217, 192]. The proposed approach achieved a recall rate of 99%. However, these studies did not classify benign and malicious bots, which is the focus of this thesis.

Shi et al. [139] proposed the use of a K-means clustering algorithm to discover malicious bots. This approach adopts a semi-supervised machine learning approach and the transition probability of the clickstream sequences. They used the CyVOD data set to evaluate their approach and achieved an overall recall rate of 98%. Chavoshi et al. [218] developed a tool for discovering bots known as DeBot, which uses the warped correlation method to analyse user activities such as tweets, retweets and liking. DeBot adopts an unsupervised learning approach that does not require labelled data to train the model and it achieved a precision of 94%.

Golbeck [219, 220] pioneered the application of Benford's law (BL) to OSN data sets. Golbeck [219, 220] demonstrated that friend_count, followers_count, and status_count obeyed BL on a human data set, whereas the same features were shown to violate BL on a malicious bot data set. Striga et al. [221] used a Facebook data set to demonstrate that BL hold likes, posts, and comments on human users. A similar finding was observed by Maurus et al. [222], namely that YouTube views, likes, dislikes, and comments conform to the BL for human users. Further research is required to validate these results.

The above studies are summarised based on data sets, significant features and performance. In the upcoming chapters, the summaries in Tables 3.3 and 3.4 are used as a benchmark against the InternetBotDetector model.

Authors	Data set type	Significant features	Performance	Approach
Oentaryo et al. [4]	Humans, benign and malicious bots	Tweet, retweet, hashtag, mention and URL	F_1 score - 74%	Supervised learning
Freeman et al. [211]	Fake and legitimate accounts	Username, email address, and IP address	AUC - 98%	Supervised learning
Cresci et al. [204]	Humans and spam bots	Not explicitly defined	Recall - 95%	Supervised learning
Gilani et al. [213]	Humans and bots	Account age, tweets, retweets, replies and mentions, URL_count, content uploaded, likes per tweet, retweets per tweet, favourites, friend_follower ratio, and activity source count	Accuracy - 48%	Supervised learning
Varol et al. [202]	Humans and malicious bots	User network patterns, activity time series, friends, sentiment, and tweet content	AUC - 94%	Supervised learning
Khaled et al. [205]	Humans and fake bots	Statuses_count, followers_count, friends_count, favourites_count, listed_count, geo_enabled, and profile features	Accuracy - 94%	Supervised learning

Table 3.3: A summary of existing methods for discovering OSN bots

Authors	Data set type	Significant features	Performance	Approach
Ruiz et al. [215]	Humans and spam bots	Retweets, replies, favorites, hashtag, URL_count, mentions, inter-time, friend_follower ratio, listed, uniqueHashtags, uniqueMentions, and uniqueURL	AUC - 89%	Supervised learning
Van der Walt et al. [203]	Humans and bots	Account age, duplicate_profile, followers_count, friends_count, geo_enabled, has_image, has_name, listed_count, status_count, and username_length	Accuracy - 87%	Supervised learning
Akyon et al. [128]	Humans and bots	Follower_count, following_count, highlight reel, external URL, tag number, and hashtag number	Precision - 91%	Supervised learning
Dorri et al. [216]	Humans and spam bots	Followers_following ratio, tweets, account age, mentions, URL_count, and spamword_count	Recall - 99%	SSML
Shi et al. [139]	Humans and malicious bots	Likes_count, comments_count, friends_count, and sharing_count	Recall - 90%	SSML
Chavoshi et al. [218]	Humans and bots	Tweets, URL_count, hashtag_count, mentions_count, and screen_name	Precision - 94%	Unsupervised learning
Wang [208]	Spam bots and non-spam	Mention_count, duplicated tweets, HTTP links, friends_count, follower_count, follower_ratio	F_1 score - 92%	Supervised learning

Table 3.4: (*Continued*) A summary of existing methods for discovering OSN bots

By examining significant features identified by the authors that focused on classifying humans and malicious bots, there was an overlap in tweets, retweets, friends, mentions, hashtags, usernames, and URL features. Tweets, retweets, and mentions refer to the number of posts, reposts, and mentions of an account, respectively. In these studies, malicious bots were observed to tweet and retweet more frequently than humans, but their accounts were less mentioned by other accounts compared to humans [4, 215].

Dorri et al. [216] found "spamword_count" to be a significant feature. The spamword_count feature counts the number of times an account mentions pre-defined spam words such as "cash prizes", "win big" and so on. More recent studies by [65, 84] argue that spammers use random words to evade detection, and it is time consuming to update a database of "known spam words". Therefore, this thesis disregarded spamword_count feature. Based on the results in Tables 3.3 and 3.4, supervised ML approaches are commonly used to discover malicious bots in OSNs. Studies by Shi et al. [139] and Dorri et al. [216] have demonstrated that the semi-supervised ML approach can produce good performance in discovering malicious bots in OSNs.

Following the discussion of related studies of bot cyberattacks in NIDSs and OSNs, the next section highlights key research gaps deduced from these studies.

3.3 Bots and cybersecurity - research gaps

This section highlights key research gaps and limitations identified by means of the SLR conducted in this thesis. The below research gaps were identified by the author of this thesis by scrutinising the findings, limitations and future research of these studies in relation to the research methods applied.

- *Benign versus malicious bot classification*: Despite the multitudes of benign bots on OSNs in particular, the majority of existing studies focused on distinguishing malicious bots from humans as indicated in Tables 3.3 and 3.4. Limited research was done to differentiate between benign and malicious bots based on the features that characterise them. From a cybersecurity perspective, the cyber threats posed by malicious bots differ from those posed by benign bots; hence, it is important to differentiate between the two types.

- *The problem of obtaining attack data*: It is evident from Section 3.1.2 that there is a scarcity of attack data in relation to training ML-based cybersecurity solutions to discover zero-day network intrusion attacks. Real-world attack data is scarce to obtain owing to the privacy laws and regulations of private data. Therefore, further research is required to address this problem.

- *Data quality checks on cybersecurity data sets*: The quality of data is a crucial factor to be considered when designing ML models, as poor data can lead to unreliable ML model outcomes. In particular, there is a growing trend in cybersecurity solutions that leverage ML and artificial intelligence (AI) models to intelligently discover cyberattacks as shown in Table 3.1. Therefore, there is a need to standardise data quality characteristics to be examined on cybersecurity data sets.

- *Discovering new unknown bot attacks*: Cybercriminals are continuously evolving their attack strategies through the use of bots, which has resulted in an increase in new attacks known as zero-day attacks. Therefore, cyber defence procedures, such as CTI, must be continuously enhanced to intelligently discover such cyberattacks.

- *Problem of describing a bot*: Given the variety of bots found in different domains in the literature, there is no clear and consistent description of a bot. Most authors characterise a bot based on its behavioural features. Moreover, the behavioural features used to describe bots differ among authors, as demonstrated in the remaining chapters.

- *Discovering a botmaster*: Although the available studies demonstrate promising results in discovering botnets, discovering a botmaster that controls a botnet network remains a challenge.

- *Enhancing CAPTCHA methods*: Recent studies demonstrated that smart bots can circumvent text-based and image-based CAPTCHA methods using AI techniques [149]; thus, new approaches need to be investigated further.

- *Accurate discovery of voice spam*: Studies on the discovery of voice spam bots are limited despite the fact that this is an important cybersecurity problem to be addressed [223].

The author of this thesis decided to address the first four research gaps as these are relevant to the research problem in Section 1.3 of this thesis.

3.4 Chapter summary

This chapter provided a comprehensive evaluation of existing studies in relation to the discovery of zero-day network intrusion and social engineering attacks in NIDSs and OSNs respectively. In addition, research gaps were identified by the author of this thesis by scrutinising the discovery methods of bot cyberattacks, findings, limitations and future research of the papers discussed. Consequently, various research gaps were identified for which the author of this thesis decided to focus on the correct classification of benign and malicious bots - given limited research on this topic. Moreover, the problem of obtaining attack data and conducting data quality checks on cybersecurity data sets were research gaps deemed by the author of this thesis to be most relevant to the research problem and questions of this thesis. Using the studies discussed in this chapter, the next chapter provides methodology requirements and develops the CySecML methodology proposed in this thesis to intelligently discover cyberattacks launched by malicious bots.

Chapter 4

The CySecML methodology requirements and development

This chapter provides a high-level description of the methodology requirements based on the studies discussed in the previous chapter. Based on these requirements, the CySecML methodology was developed that provides organisations with a set of guidelines for developing machine learning based cybersecurity solutions. Thereafter, the CySecML methodology is used to address the research problem of this thesis, which is to intelligently discover cyberattacks launched by bots on Internet Application Platforms (IAP). The methodology requirements are discussed next, followed by the development of CySecML methodology.

4.1 Methodology requirements for building machine learning based cybersecurity solutions

4.1.1 Related work and background

In addition to the studies identified through the systematic literature review in Chapter 3, the author of this thesis conducted a high-level review of existing cybersecurity methodologies that are used to intelligently discover cyberattacks on IAP. The aim is to assess state-of-the-art methodologies in terms of their strengths and limitations, and how they can be enhanced to address some of the research gaps discussed in Chapter 3. To recap, some of the identified

research gaps include the problem of obtaining or creating attack data, and conducting data quality checks on cybersecurity data sets. These methodologies are summarised next.

4.1.1.1 Summary of existing cybersecurity methodologies

Dekker et al. [224] proposed a cybersecurity methodology called threat intelligence based security assessment (TIBSA) that aims at providing guidelines for intelligent discovery of cyber threats. The TIBSA methodology leverages casual graph models rather than ML algorithms to make informed predictions about cyber threats. Casual graph models are statistical methods that do not use machine learning - this approach does not meet the scope of this thesis, and is thus not considered. Noorizadeh et al. [225] proposed a data-driven cyberattack detection (CAD) methodology. Their methodology consisted of three key blocks: data collection from the Web, data normalisation, and prediction using ML algorithms. In the context of this thesis, the Web is an IAP, data normalisation is part of data cleaning and feature selection, and prediction is part of the machine learning and decision-making blocks in the methodology requirements presented in Figure 4.1. The CAD methodology is not suitable for this thesis because it does not provide guidelines for obtaining or creating data for the training of complex attacks such as zero-day network intrusion attacks.

Coulter et al. [226] proposed a data-driven cybersecurity (DDCS) methodology for discovering cyberattacks, which was tested $\mathbb{X}$ data set. The DDCS is based on three main blocks: data processing, feature engineering, and modelling. The blocks in DDCS are common in most ML-based cybersecurity solutions; however, data quality checks and obtaining attack data guidelines are not included. The same limitations were observed on the below methodologies such as Spanos et el. [227] - who proposed an anomaly based methodology for discovering cyberattacks on IoT environments. Their methodology consisted of data collection, feature selection, data normalisation, and ML algorithms. Datta [228] proposed a cybersecurity framework to discover cyberattacks in IoT environments. Datta [228] methodology consists of security incident and event monitoring (SIEM), risk assessment (risk identification, estimation and evaluation) and cyberattack resilience. Parlapalli et al. [229] proposed an anomaly based cybersecurity methodology for discovering cyberattacks on NIDS using ML or AI algorithms. Their methodology includes data collection, data cleaning, feature selection, ML, and result blocks. Magnani et al. [230] proposed a ML-based cybersecurity methodology for enhancing NIDS to discover cy-

berattacks. Data extraction, feature selection, and threat detection based on anomaly detection are the key blocks of their methodology. Arnau et al. [231] proposed an anomaly based cybersecurity methodology for discovering cyberattacks in IoT environments. Their methodology consisted of data collection, data pre-processing, data cleaning, ML or AI models and evaluation blocks.

Existing methodologies, such as those mentioned above, lack a structured approach for obtaining cybersecurity data sets particularly for the purposes of discovering complex cyberattacks such as zero-day network intrusion attacks, and the data quality aspects that need to be considered when designing ML-based cybersecurity solutions. Hence, this thesis proposes the CySecML methodology to fill this gap.

4.1.2 Methodology requirements for building intelligent systems for discovering malicious bots

The methodology requirements presented in this section are common in many ML and AI based cybersecurity solutions such as data extraction, feature selection and machine learning algorithms. For example, Horowitz et al. [232], Sarker et al. [37] and Khder et al. [78] among others. The aim for presenting Figure 4.1 is to demonstrate how the design of existing (discussed in the previous section) intelligent decision-making ML-based cybersecurity solutions can be enhanced by incorporating concepts such as synthetic data and data quality checks.

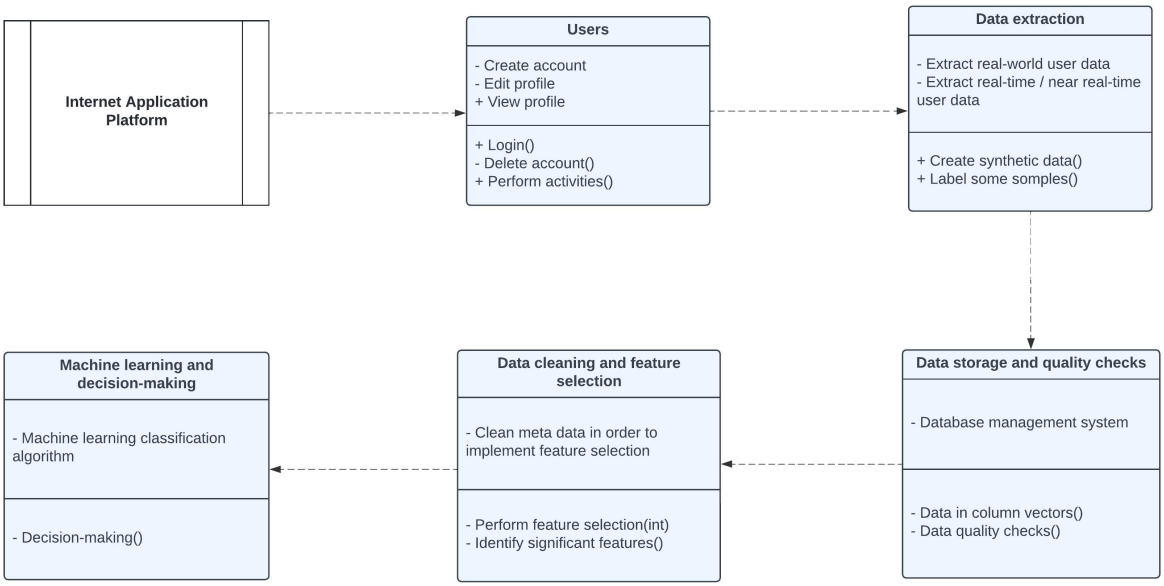


Figure 4.1: High-level methodology requirement using UML class diagram

Standard UML class diagram symbols were used in Figure 4.1 for example, "+" symbol represent a public attribute and "-" a private attribute on the class operators.

Internet Application Platform: This block describes an IAP such as 𝕏, which aims to implement a cybersecurity solution to intelligently discover malicious bots to ensure that the CIA triad of cybersecurity is adhered to.

Users: This block describes the users of an IAP which are assumed to include bots in this thesis. In this instance, users can create an account, view profile, and edit profile among other activities. Consider 𝕏 as an example: a user is required to create an account that is authenticated by the system controllers of that IAP prior to the user being permitted to use that IAP.

Data extraction: This block deals with the extraction of user data from an IAP. This data is used to discover users displaying anomalous behaviour. Data extraction includes collecting real-world data and creating synthetic data in real-time or near real-time. In this current study, the author of this thesis used publicly available data sets, as described in Chapter 5. Various data extraction tools exist, such as the CICFlowMeter tool [233] which is used to capture and extract packet information in a network in a column vector format. Extracting data in a column vector format is generally considered a good practice for storage and presentation quality [10, 67, 77].

Data storage and quality checks: This block deals with the storage of data extracted from the previous block. Ideally, data should be stored in a cloud based database storage system that can store big data [77]. Data storage was not covered in this thesis, as the data sets downloaded were stored in the local drive. In addition, the extracted data is examined to determine whether it is of quality. The data quality characteristics of cybersecurity data sets are discussed in Chapter 6.

Data cleaning and feature selection: The data cleaning component involves correcting data errors, such as blank fields, commonly referred to as data corruption. Data corruption may occur at any stage in a system between the reading, processing, and storage of data [234]. In addition, this block includes the conversion of non-numerical data to numerical data and data normalisation because feature selection is applicable to numerical data. Data normalisation is a process of removing outliers in a data set. In this thesis, only features with numerical data were considered for the purposes of implementing feature selection; therefore, the task of converting data was not required. Once the data is cleaned, feature selection can be implemented to identify significant features for use as inputs in a machine learning classification model. The feature selection process is discussed in detail in Section 7.2.1.

Machine learning and decision-making: This block deals with the implementation of a classification machine learning (ML) algorithm that makes a prediction, i.e., decision-making whether a bot activity is malicious or not. For example, a ML classification algorithm can be of supervised, unsupervised, and semi-supervised types. Section 7.2.4 provides details of the different types of ML algorithms, and this thesis adopts a semi-supervised ML algorithm.

The methodology requirements are discussed in this section. In particular, the presented methodology requirements aim to address the current challenges in ML-based cybersecurity solutions, that is, the scarcity of obtaining real-world attack data, and ensuring that the cybersecurity data sets used to build intelligent ML-based systems produce reliable results. These requirements are now used to develop the CySecML methodology that provides specific guidelines for designing ML-based cybersecurity solutions for the intelligent discovery of malicious bots on IAPs. This is discussed in the next section.

4.2 The CySecML methodology development

4.2.1 Introduction

The previous section introduced the methodology requirements for designing intelligent machine learning based cybersecurity solutions. Using these requirements, this section now develops a methodology called CySecML that provides organisations with a framework for developing machine learning based cybersecurity solutions. Specifically, the CySecML methodology provides a structured approach that can guide organisations to ensure that important aspects such as extracting cybersecurity data sets, checking data quality and feature selection are adequately implemented, particularly for discovering complex attacks such as zero-day network intrusion attacks. The development of the CySecML methodology is discussed next, followed by its pseudo-code representation.

4.2.2 Related work and background

According to the author of this thesis, the existing cybersecurity methodologies discussed in the previous section, do not provide organisations with guidelines for obtaining or creating cybersecurity data sets, and data quality aspects that should be considered when designing ML-based cybersecurity solutions. Often, organisations do not have access to real-world attack data, such as zero-day network intrusion attacks, so the question becomes, how can they effectively train their cybersecurity solutions to effectively discover such attacks? In addition, what data quality aspects should be considered so that one can trust the results obtained. The CySecML methodology aims to address these questions.

4.2.3 Experimental work

4.2.3.1 The CySecML methodology development

Using the methodology requirements in Figure 4.1, the CySecML methodology is developed that consists of data preparation and InternetBotDetector model components. Data preparation component consists of data extraction, data storage and quality checks as well as data clean-

ing and feature selection blocks. InternetBotDetector model consist of machine learning and decision-making block. User (assumed to be a bot) behaviour analysis methods are used to analyse user activities [213]; for instance, the frequency and times at which a user posts content in OSNs, so that normal and abnormal behaviours can be discovered.

CySecML is a data-driven methodology that optimises existing techniques that include data quality checks, feature selection and machine learning to intelligently discover cyberattacks launched by bots on IAPs. Let us consider 𝕏 as an example, where user activities include the number of posts and friends an account has at a particular point in time. The identified significant features are used as inputs to the InternetBotDetector model. Figure 4.2 below depicts the key steps involved in the CySecML methodology.

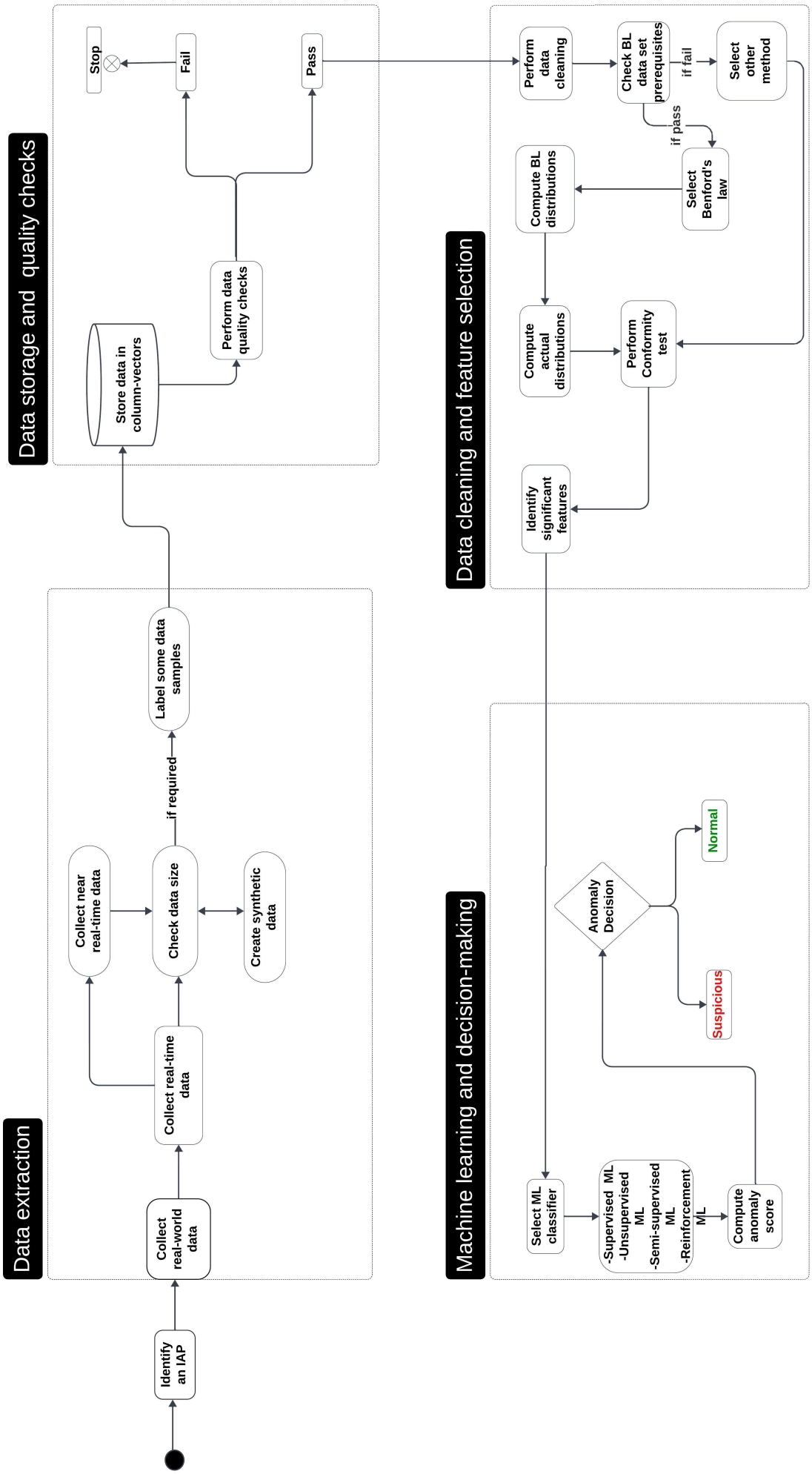


Figure 4.2: High-level design steps of the CySecML methodology

The CySecML methodology is applicable to IAPs such as website, NIDSs and OSNs among others, where the discovery of anomalies is of paramount importance from a cybersecurity perspective. This methodology is unique from existing cybersecurity methodologies given its unique components on data extraction, data quality checks and Benford's law properties to identify significant features in cybersecurity data sets. The details of the CySecML methodology steps are provided below.

Identify an IAP: The first step is to identify a suitable IAP where the CySecML will be implemented and its users that can be people, bots or organisations. People and organisations form part of cyberspace elements, therefore, protection of humans from cybercrimes is of crucial importance from a cybersecurity perspective. For instance, consider users of 𝕏 that can be broadly categorised as either human or bot. In practice, it is sometimes difficult to distinguish between human and bot cybercrimes [16, 83]. For instance, Van der Walt [141] studied identity deception in OSNs launched by humans using features such as "dup_profile", which determines whether an account has a profile description like that of other accounts. On the other hand, Cresci et al. [117] studied identity deception launched by bots in OSNs by examining their tweets and retweet features, whereby these bots were observed to mimic the identity of famous people.

Collect real-world data: This step involves collecting real-world data from an IAP that is used to discover anomalies. ML-based cybersecurity solutions should be trained and tested on a real-world data set such that the solutions used can produce trusted results.

Collect real-time data: This step relates to instantaneous user data collected from an IAP. State-of-the-art data-driven cybersecurity solutions depend on real-time data to effectively discover cyberattacks. [78]. Data such as attributes and features are used to describe users [163, 235], for example, "Destination_IP address" feature that describes the IP address of the server to which network traffic is forwarded [8].

Collect near real-time data: This step deals with collecting near real-time user data because collecting real-time data can be challenging in some instances. For example, network connectivity issues may affect the availability of real-time data.

Check data size: This step checks the size of the collected data. ML algorithms should ideally be trained on a sample size ten times larger than the number of features or attributes [236]. For example, if a data set contains five features, then the reasonable sample size should be at least 50. If the collected real-world data is not sufficient as per the above, then an alternative would be to supplement it with synthetic data.

Create synthetic data: If an organisation is not in possession of sufficient real-world data, then an alternative would be to consider obtaining synthetic attack data. There exist various tools such as the CICFlowmeter [233] tool used to generate network traffic data. ML-based cybersecurity solutions such as the InternetBotDetector model rely on the availability of cybersecurity data.

Label some data samples: This step deals with the labelling of data samples. This step depends on the choice of a ML algorithm used, i.e., supervised - requires labelled data, unsupervised - does not require labelled data, semi-supervised - requires some labelled data and reinforcement - does not require labelled data.

Store data in column vectors: This step deals with the storing of collected data. For ML algorithms, the data set is preferred to be stored in column vectors mainly to simplify the implementation of feature selection [230, 237].

Perform data quality checks: This step aims to ensure that the data sets used to train ML-based cybersecurity solutions are of quality, and suitable for cybersecurity problems. ML-based cybersecurity solutions should be trained and tested on trusted and reliable data sets so that solutions produce dependable results. The data quality characteristics to be checked include the availability, usability, reliability, relevance, data size and presentation quality of cybersecurity data sets. If a cybersecurity data set fails to meet all these data quality checks with an exception of presentation quality, one should not proceed further, because training a cybersecurity ML algorithm using unreliable data will result in unreliable outcomes [238, 236]. Unreliable data can lead to high false alarm rates in cybersecurity domains such as NIDSs.

Perform data cleaning: This step deals with the cleaning of the collected data. This step is usually completed by a cybersecurity analyst or system controller of an IAP. Cybersecurity data from an IAP is not always clean for the purposes of building ML-based cybersecurity solu-

tions. For instance, cybersecurity data may contain corrupt or invalid information such as blank fields on features that may occur during the processing of data [234]. Moreover, specifically for this thesis, negative values were removed so that BL could be applied to real positive numbers as a feature selection method. For example, features such as "FlowSentRate" found in the CIRACICDoHBrw2020 data set, which had some negative values, were removed.

Check BL data set prerequisites: This step deals with the minimum data set prerequisites that must be met to correctly use BL. Not all cybersecurity data sets from IAPs are expected to conform to BL, therefore, an assessment must be conducted to determine whether a cybersecurity data set meets BL data set prerequisites. For example, one of the BL data set prerequisites is that leading digits 1 to 9 must be observable. This is further discussed in Chapter 7. If all BL prerequisites are satisfied, then one can select BL as a feature selection method; otherwise, a different feature selection method can be selected.

Compute BL distributions: This step involves calculating BL distributions based on the properties described in Chapter 7. The five BL distributions are discussed in Chapter 7 - and this forms part of the building blocks for identifying significant features indicative of anomalous behaviours on cybersecurity data sets.

Compute actual distributions: This step deals with the computation of cleaned (actual) distributions of leading digits, following the previous block.

Conformity test: This step deals with the comparison of BL versus actual distributions of leading digits. Using statistical methods such as the Pearson chi-squared distribution test is conducted using 95% confidence level.

Identify significant features: This step is the output or results of the data preparation component, which is subsequently used as an input into the InternetBotDetector model component. The importance of identifying significant features prior to implementing ML-based cybersecurity solutions is crucial and well understood in the cybersecurity community - as this can ultimately impact the overall performance and computational cost of a cybersecurity ML algorithm.

Select ML classifier: This forms part of the InternetBotDetector model component and deals with the ML algorithm(s) implemented to intelligently discover cyberattacks on IAPs. In recent

years, ML has advanced the discovery pillar of cybersecurity frameworks by enhancing standard monitoring techniques with intelligent ML-based solutions. Supervised, unsupervised, semi-supervised and reinforcement are ML algorithms that can be considered. Moreover, a combination i.e., ensemble of ML algorithms is permissible. Specifically for this thesis, semi-supervised ML (SSML) algorithms are implemented to discover cyberattacks launched by bots.

Compute anomaly score: This step forms part of the InternetBotDetector model component and computes an anomaly score based on precision, recall, F_1 score and MCC score.

Anomaly decision: This is the output of the InternetBotDetector model component, where a decision is taken about whether a specific bot activity is malicious or normal. For illustrative purposes, InternetBotDetector deems an activity to be malicious if either the MCC or the F_1 score exceeds 80%, but considers it normal if these scores are below the 50% threshold. A self-adaptable threshold can be considered if the system controllers choose this approach. If a bot activity is flagged as being suspicious or inconclusive, the IAP system controller or cybersecurity analyst can block that account and conduct further investigations if required. This is a standard process for CTI procedures when dealing with suspicious activities on IAP [38, 41].

Having described the components of the CySecML methodology, the next section provides the pseudo code representation of the CySecML methodology to illustrate its implementation.

4.2.3.2 The CySecML methodology - pseudo code representation

The blocks involved in the CySecML methodology were discussed above. However, these blocks do not provide guidelines for the real-world implementation of this methodology. To fill this void, a pseudo code representation is usually required [239, 240]. The CySecML methodology steps were derived in Figure 4.2. To recap, the are four key blocks, block I - data extraction, block II - data storage and quality checks, block III - data cleaning and feature selection and block IV - machine learning and decision-making. The aim of a pseudo code representation is to explain the implementation steps of this methodology in a less technical manner.

Algorithm 1: The CySecML methodology

Block I - data extraction

Select an IAP and its users $U = \{u_1, u_2 \cdots, u_n\}$. Define $F = \{f_1, f_2 \cdots, f_m\}$ as a set of features from IAP$_i$ that describe activities of U. Collect real-time data D from an IAP, check data size and add synthetic data if required.

Block II - data storage and quality checks

if *D passes data quality checks* **then**
| perform data cleaning

else
| stop the process

end

D is a union of X_L and X_U where

$X_L = \{x_1, x_2 \cdots, x_m\}$ is a set of labelled data points of set U.

$X_U = \{x_1, x_2 \cdots, x_m\}$ is a set of unlabelled data points of set U.

Block III - data cleaning and feature selection

if *D passes BL data set prerequisites* **then**
| select BL as a feature selection method

else
| select other method

end

For all $f_i \in F$ compute BL distributions on set X_L;

if *f_i obeys one of the BL distributions on a normal data set while violating all BL distributions on a malicious data set* **then**
| add f_i to a subset $F' \subseteq F$

else
| disregard f_i

end

Block IV - machine learning and decision-making

Let $h(\theta)$ be a machine learning classifier, SSML in this thesis

(i) **For all** $U \in X_U$, label X_U using $h(X_L|F')$;

(ii) Compute anomaly score $\kappa \in [0, 100]$ using $h(X_L|F')$;

if $\kappa \geq 80\%$ **then**
| label u_i as a malicious behaviour

else
| label u_i as a benign behaviour

end

The CySecML methodology in algorithm 1 aims to provide a stepwise guidance of implementing this methodology.

4.3 Chapter summary

This chapter formally presented the CySecML methodology - including its pseudo code representation. The CySecML methodology enhances existing ML-based cybersecurity methodologies by incorporating techniques that include data collection, data quality checks and Benford's law - as a feature selection method. These techniques have been proposed to enhance the intelligent discovery of cyberattacks, such as the social engineering attacks launched by bots. The main shortfall of the CySecML methodology is that some of the steps, such as data quality checks, data cleaning, BL prerequisites, and data labelling, require human judgement. The attempt to automate these steps is beyond the scope of this thesis. Having developed the CySecML methodology, the next chapter deals with the data extraction block to provide proof-of-concept for this methodology.

Chapter 5

Cybersecurity data sets

5.1 Introduction

To provide proof-of-concept for the CySecML methodology introduced in the previous chapter, cybersecurity data was required for illustrative purposes. The question now is how to obtain or create attack data that can be used to train machine learning based cybersecurity solutions, this chapter addressed this question. The concept of big data is introduced first, as it is a common phenomenon in cybersecurity data sets and is a crucial factor to consider when designing data-driven discovery or detection cybersecurity solutions. Figure 5.1 depicts the focus of this chapter.

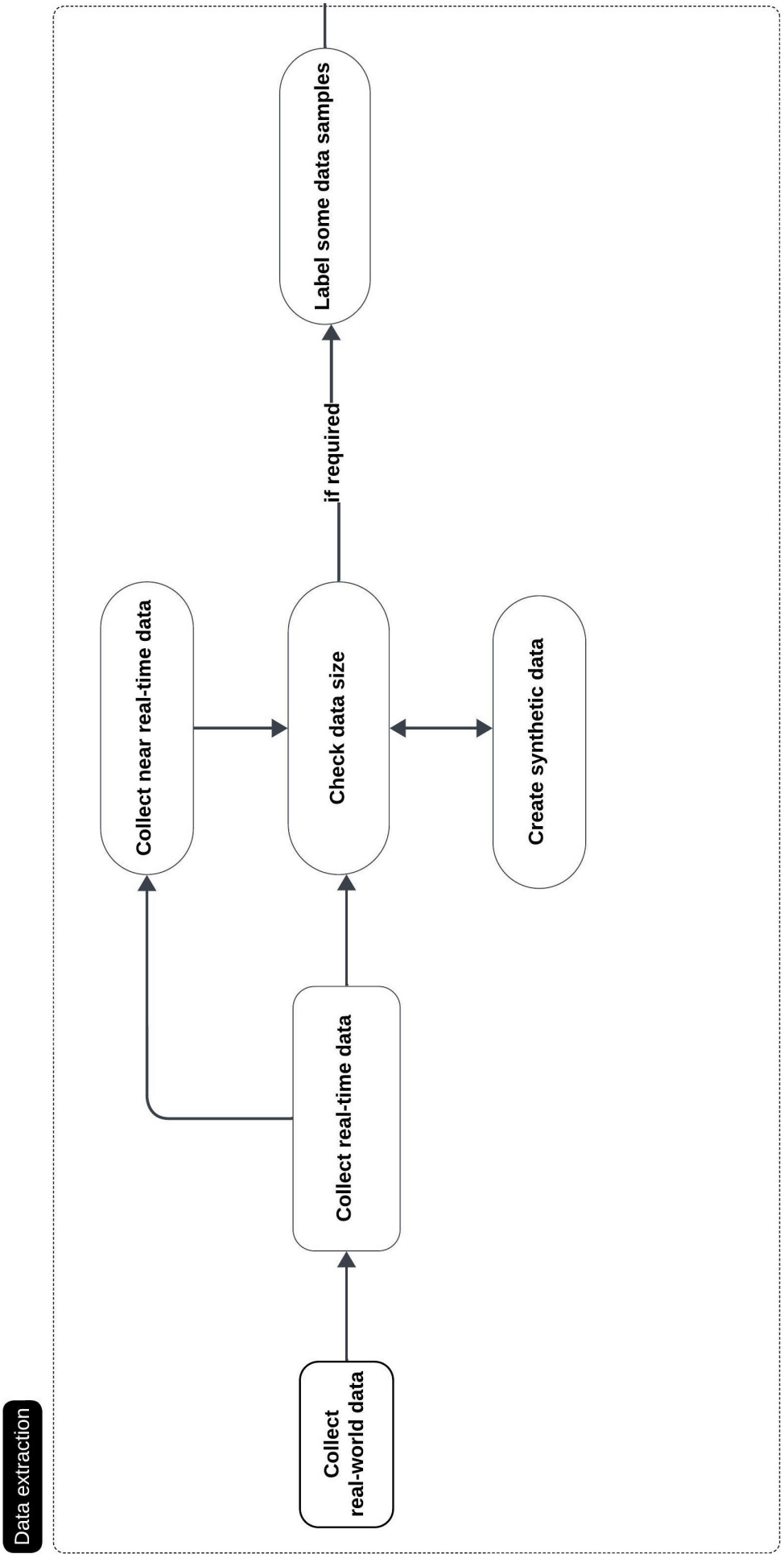


Figure 5.1: Data extraction - taken out from Figure 4.2

5.2 Related work and background

5.2.1 Big data

Big data has become a fundamental research problem [241] owing to various real-world application domains, such as computer networks. Computer networks are affected by high volumes of network traffic, various network types, and the velocity of network traffic [242, 67]. Big data is characterised by these main "3Vs", namely high volume, high variety, and high velocity [67]. High volume refers to large amounts of data [241]. For example, millions of tweets are observed each day on 𝕏, thus large data storage devices [243] to store this data. High-variety refers to a diverse data set [241]. For example, various types of bots found on OSNs, such as spam bots, news update bots, and download bots among many others. High velocity refers to the speed at which data is generated [241]; for instance, the rate at which network traffic instances are generated in a wireless network. In the context of this thesis, Internet Application Platforms such as online social networks (OSNs) and network intrusion detection systems (NIDSs), are widely considered big data platforms [76, 67].

Big data presents several challenges for cybersecurity solutions based on machine learning that are used to discover cyber threats on OSNs and NIDSs [76]. Some of these challenges are listed below:

- There are multitudes of features on OSNs and NIDSs [244] that are used to describe user information or activities. With such a variety of features that can be considered as inputs to machine learning algorithms for the purpose of intelligently discovering cyber threats, the question becomes - which features (if any) are important and what is their impact on computational cost?
- Often, the number of instances observed between benign and malicious activities on cybersecurity data sets such as OSNs and NIDSs is disproportional [76]. In most cases, benign activities are significantly more common than malicious ones, also referred to as class imbalanced. Thus, aiming to discover activities of a minority class, i.e., malicious activities, impose challenges on machine learning algorithms [245].

Next, the presence of class imbalance in cybersecurity data sets is discussed in more detail in relation to the intelligent discovery of bot cyberattacks on OSNs and NIDSs.

5.2.1.1 High-dimensional imbalanced data sets

Introduction

The cybersecurity data sets used in this thesis to intelligently discover bot cyberattacks on OSNs and NIDSs are considered imbalanced, as demonstrated later in this chapter. Imbalanced or skewed data sets refer to cases in which the distribution of the number of instances for one class is significantly greater than that of another class or classes [242, 245]. For example, consider the UNSW-NB15 NIDSs data set that contains a large amount of benign network traffic (N_B) - 56000 and a small amount of malicious network traffic (N_M) - 130 (worm attack), i.e., $N_B \gg N_M$ or vice-versa. Imbalanced data sets are prevalent in many real-world applications such as credit card fraud detection and network intrusion detection [242, 181]. A high dimension refers to the large number of features or attributes that describes a data set [242], for instance, the IoT Intrusion-2020 data set has 86 features. High dimensionality often increases the computational cost of a machine learning (ML) algorithm [242, 246]. Various techniques have been developed to address class imbalances, particularly when building ML algorithms. ML algorithms such as those that employ supervised algorithms can suffer bias towards the majority class if the data sets are imbalanced [35].

Challenges of high-dimensional imbalanced data sets

The challenge of high-dimensional imbalanced data sets is generally mitigated by employing dimensionality reduction techniques prior to training an ML algorithm [242, 76, 172]. Dimensionality reduction refers to the methods that aim to reduce high-dimensional input features to lower dimensions [242]. Feature selection and extraction are two primary methods for reducing high-dimensional input features [246, 247]. Feature selection reduces the high-dimensions by identifying only a sub-set of important or significant features using a specific technique [246, 247, 248]. On the other hand, feature extraction is a method that reduces high-dimensions by projecting a high-dimensional set of input features into a lower set of features by preserving as much information as possible [246, 247]. The principal component analysis (PCA) is a

classic example of a feature extraction method [76]. Most authors have used feature selection and extraction terms interchangeably without any impact. In this thesis, feature selection was implemented prior to training the ML model proposed to intelligently discover cyberattacks launched by bots.

The next section discusses how to obtain cybersecurity data sets for the purposes of training ML-based cybersecurity solutions.

5.2.2 How to obtain or collect data for ML-based cybersecurity solutions?

Cybersecurity solutions such as the InternetBotDetector model, which employs ML algorithms to discover bot cyberattacks in IAPs, depend on the data used to train them [249]. For instance, the predictive power of the InternetBotDetector model in discovering zero-day network intrusion attacks depends on the data used to train it, i.e., attack data. In such a scenario, the model must be trained on a data set that will enable it to correctly discover zero-day attacks. In practice, obtaining real-world attack data is burdensome, owing to the sensitive and private nature of attack data [250]. In the case of this research, obtaining data on zero-day network intrusion attacks launched by bots was difficult; in fact, the author of this thesis found that none was available. It may be possible to obtain real-world data of particular attacks from organisations that fell victims to such attacks [251], see the common vulnerabilities and exposure (CVE) [1] program that identifies publicly disclosed cybersecurity vulnerabilities with their data records. However, this research was not supported by any organisation that could provide real-world attack data; hence, alternative methods were sought and so-called synthetic attack data was used for the purposes of training the InternetBotDetector model.

5.2.2.1 Synthetic data

To overcome the availability of large real-world attack data and security concern challenges, researchers have introduced *synthetic data* [252]. From a cybersecurity perspective, this is referred to as synthetic attack data that is generated in a controlled environment such as a laboratory using data-generating tools and rules extracted from a real-world data set [250, 253]. According to Dankar et al. [253], the concept of synthetic data was originally proposed by

[1] https://cve.mitre.org/. Last accessed: 29Apr2024.

Rubin in 1993 [254]. Since then, multiple researchers have worked on this topic. For example, Bowles et al. [250] proposed using the IBM data fabrication platform to generate synthetic cancer data, while Kim et al. [255] proposed using the building information model (BIM) to generate synthetic image data. ML algorithms such as neural networks [250] and deep learning (DL) that include generative adversarial networks [256] can also be used to generate synthetic data sets.

It is crucial that synthetic data depicts the true characteristics of real-world data so that one can trust the results produced by an ML algorithm trained on synthetic data [250, 253]. The challenges mentioned can be mitigated as one can generate synthetic attack data that is large enough to meet the need. Furthermore, because synthetic attacks are generated in a controlled environment, there are no security or privacy concerns, as these data sets contain no personally identifiable information (PII) [253]. Thus, synthetic data sets do not pose any cybersecurity risks when shared with the public [253]. According to Singh et al. [59], privacy and security concerns are the main reason why organisations do not publicly share their real attack data. They are concerned that this may result in more attacks if cybercriminals believe that their security systems are vulnerable. Anonymisation, a method of removing PII information from a data set, is a technique that can be considered to overcome security and privacy concerns [253]. However, it does not address the challenge of real-world attack data that is not sufficiently large for training an ML algorithm [250, 257]. Thus, in most cases data synthetisation is preferred over data anonymisation for the purposes of training an ML-based cybersecurity solution [253]. Noting that various synonyms are used to refer to synthetic attacks including “simulated attacks” [2], “artificial attacks” [256] and “fabricated attacks” [258].

Although synthetic data is valuable to train ML-based cybersecurity solutions, it is not a replacement for real-world data [250]. The main advantage of using synthetic data is that one can generate large-scale data sets from real-world data sets and start training ML algorithms [257, 256]. In addition, large-scale data overcomes the problem of overfitting, which is generally associated with real-world attack data, i.e., small samples [250]. One can analyse this data and extract rules using synthetic data generators [250, 255]. These rules include the type of data (e.g., numeric, non-numeric, Boolean), range of values, relationships among features of the data sets, and so on [250]. In addition, a user of these tools can manually add rules to be included for the generation of synthetic data if the rules cannot be easily extracted from real-world data [250, 255].

Formally, data synthetisation can be described as follows:
Consider a real-world data set X containing n_X samples extracted from a population P (e.g., IAP). Each row x^i is a data record described by a set of features $F = \{f_1, f_2, \cdots, f_m\}$. The goal of synthetic data is to take the data set X as an input, and construct a statistical model that captures statistical properties of P such as the standard deviation quantile. Next, using a statistical model, a synthetic data set X_S can be generated that is statistically similar to X. This description was adopted from [253, 254]. For this research, the specific question is how to best train the InternetBotDetector model to effectively discover zero-day network intrusion attacks.

According to Ahmad et al. [188] it is practically impossible to train an ML-based NIDS to discover every type of new attack, i.e., zero-day attack. However, several approaches can be considered for the purposes of adequately train an ML-based NIDS to discover zero-day attacks. For instance, multitudes of network intrusion data sets exist in the literature, see a summary of these data sets in [188, 259, 57]. In general, and particularly referring to the network intrusion data sets used in this thesis, the attacks considered in these data sets were synthetically generated. However, as previously mentioned, they do not contain zero-day, i.e., unknown attacks. A known attack is publicly disclosed and known by the cybersecurity community (even the general public), whereas an unknown attack is neither publicly disclosed nor known by the cybersecurity community [36, 188]. In this thesis, a known attack is an attack that is found in a particular data set, with its label, whereas an unknown attack is not part of a particular data set [260]. Now the question is, how to create a zero-day type of attack from known attacks in a data set?

Synthetic data for complex zero-day attacks

The rule of thumb in creating a zero-day attack type from known attacks is to not use all known types of attack in the training and testing stages of an ML algorithm [175, 179, 188]. Using all available network intrusion types of attack in both the training and testing phase of the ML algorithm disregards the presence of a zero-day attack [188]; thus this approach is not preferred. According to Ahmad et al. [188], the following are the two most common approaches for creating a zero-day attack type from known attacks:

- Approach 1: Randomly use different types of attack in the training and testing stages of

an ML algorithm.

- Approach 2: Combine all known attacks from a data set and create a new zero-day attack type that is based on known types of attack.

Approach 1

The first approach for creating a zero-day type of attack for illustrative purposes is to train and test a ML algorithm using different types of known attacks [178, 175]. An attack type that appears for the first time in the testing phase of an ML algorithm but not in the training phase can be deemed a "new unknown", i.e., zero-day attack type [178, 175]. The challenge with this approach is how best to choose types of attack to use in the testing phase as zero-day attack types. Authors such as Zoppi et al. [178] , Pu et al. [177] and Pallaprolu et al. [169] who adopted this approach, randomly selected different type of attacks to use in the testing phase as zero-day attack type. As far as the author of this thesis is concerned, there is no scientific evidence in the literature on choosing the "optimal" combination of types of attack to use as zero-day attack type. Consider for example the IoT Intrusion-2020 data set that contains eight malicious network traffic types, one can have a minimum of : $\frac{8!}{2!(8-2)!} = 28$, possible combinations of zero-day attack types. Thus, the author of this thesis does not consider this approach feasible.

Approach 2

The second approach is to manually create a new unknown or zero-day attack type by combining known attack types of a particular data set [260, 188]. In this way, a zero-day attack type is created from known types of attack - an approach referred to as creating *known unknowns* [188, 261, 262]. The premise of this approach is that a network traffic data set is based on the same description of features and structure; thus, one can combine different known types of attack across the same features. Authors such as Chiba et al. [160] and Zavrak et al. [175] adopted this approach of manually combining all known types of attack to create a new unknown type of attack. The author of this thesis deemed this approach feasible to implement, as a zero-day attack type is easily created by combining known attack types, and therefore, opted for it as the base case for creating a zero-day attack type. This approach of creating a zero-day attack type is dependent on a data set that has a clear label of known attack types [188, 175]. By combining known types of attack, the sample size of a zero-day attack type should be large enough if individual attacks are also large. In this way, over-fitting problems are avoided [188].

Figure 5.2 summarises the steps suggested in "Approach 2" for creating synthetic attack data, using the UML sequence diagram. This will form part of data extraction block in the CySecML methodology in Figure 4.2.

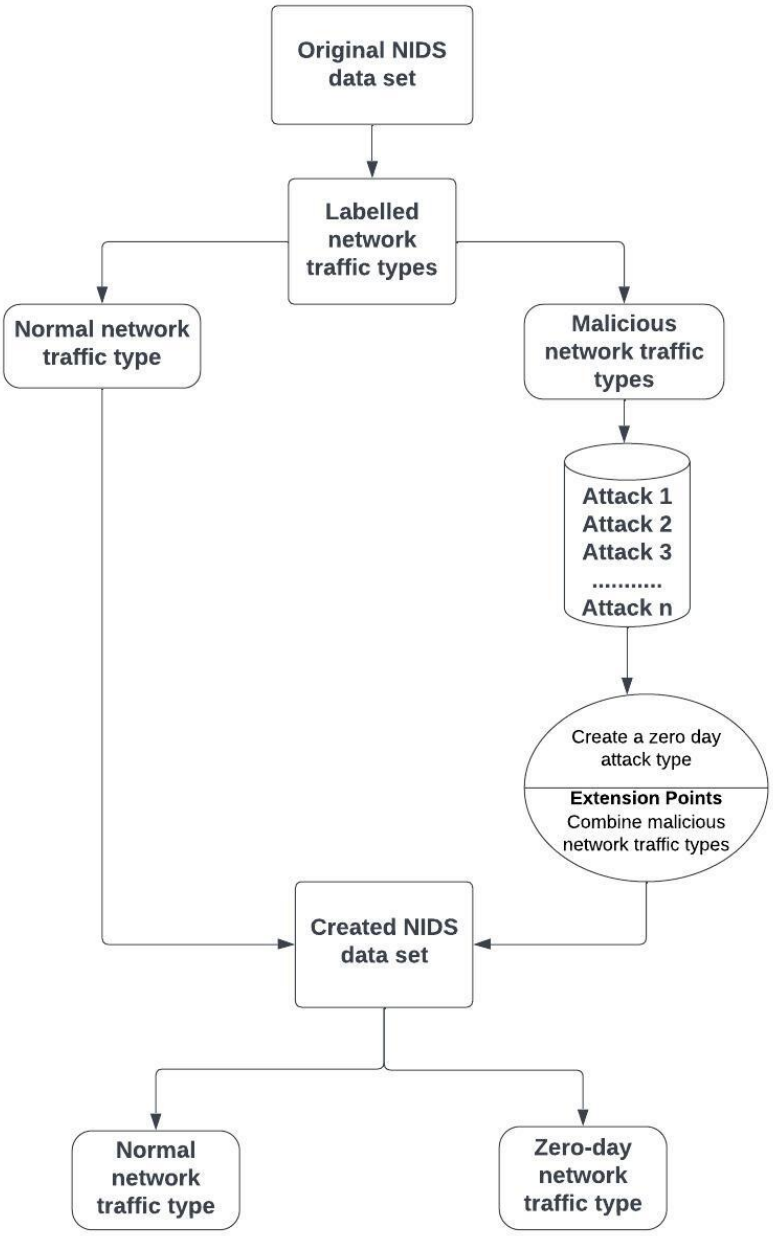


Figure 5.2: Steps for creating a zero-day attack type from an existing NIDS data set

In both approaches above, it is possible to quantitatively evaluate the performance of the model [188], as a zero-day attack type is in theory known beforehand.

5.3 Experimental work

As indicated in the introduction of this chapter, this thesis focuses on discovering zero-day network intrusion attacks in NIDSs. The next section deals with the data extraction step in Figure 5.1. To discover zero-day network intrusion attacks in NIDSs, the IoT Intrusion-2020, UNSW-NB15, CICDDOS2019 and CIRACICDOHBRW2020 cybersecurity data sets were used for illustrative purposes in this thesis.

5.3.1 NIDSs cybersecurity data sets

The author of this thesis conducted a high-level assessment on existing network intrusion attack data sets published from 2015 and beyond, see [259, 188]. This was done to select four cybersecurity data sets containing bot related cyberattacks. Four popular network intrusion data sets were selected from a number of data sets to create variations in terms of old (i.e., 2015 to 2019) and recent (2020) network traffic patterns [164]. The IoT intrusion-2020 data set is introduced next.

5.3.1.1 IoT Intrusion-2020 data set

Ullah et al. [8] developed a network intrusion data set called IoT Intrusion-2020, based on synthetic attacks. This data set was developed in 2020 based on various synthetic botnet attacks (including flooding attacks) and benign network traffic. In addition, the authors of this data set used the CICFlowMeter tool to generate network traffic data. The CICFlowMeter [233] is an open-source tool used to generate bidirectional network traffic flows, such as source-to-destination ports, and vice versa. The CICFlowMeter tool has more than 80 bidirectional features that include the number of packets amongst others, see Figure 5.4. Table 5.1 highlights the network traffic of the IoT Intrusion-2020 data set.

Network traffic	**Description**	**Number of instances**
Benign	Normal network traffic.	40073
SYN Flooding	"The attacker attempts to cause a network malfunction by setting up multiple half-open connections without finalising them."	59391
UDP Flooding	"The attacker attempts to overwhelm random ports in a network so that they will not be available to legitimate users."	183554
HTTP Flooding	"The attacker attempts to manipulate the normal HTTP with a malicious one to attack a Web server."	55818
ACK Flooding	"The attacker attempts to overwhelm the network server with ACK packets."	55124
Host Brute force	"The attacker attempts to guess the password of a host by submitting numerous passwords."	121181
ARP Spoofing	"The attacker sends manipulated Address Resolution Protocol (ARP) to illegally connect over a network."	35377
Scan host port	"The attacker attempts to gather information about the device by issuing multiple hotspots."	22192
Scan port OS	"The attacker observes a network to figure out open ports that can be exploited."	53073

Table 5.1: The IoT Intrusion-2020 network traffic types and descriptions adopted from [8]

Table 5.1 presents - in a summary format - synthetically generated cyberattacks in a laboratory using bots as stated by [8]. In addition, the general descriptions of these types of attacks were extracted from [8]. Benign network traffic is labelled as "normal", while malicious network traffic is labelled as "anomaly" [8]. For the purposes of this thesis to discover zero-day network

intrusion attacks, the author of this thesis manually created a new zero-day attack by combining anomalous network traffic in Table 5.1, following the steps outlined in Figure 5.2. Specifically, a zero-day attack was created by combining SYN Flooding + UDP Flooding + HTTP Flooding + ACK Flooding + Host Brute force + ARP Spoofing + Scan host port + Scan port OS across the same features.

Figure 5.3 below illustrates a sample of the zero-day attack created by the author of this thesis.

Flow_Duration	Tot_Fwd_Pkts	Tot_Bwd_Pkts	TotLen_Fwd_Pkts	TotLen_Bwd_Pkts	Label
120	1	1	30	1388	Benign
73	0	2	0	0	Benign
367	0	2	0	1388	Benign
451	0	3	0	4164	Benign
382	1	1	1388	1388	Benign
385	0	2	0	0	Benign
393	1	1	0	0	Benign
79	0	2	0	0	Benign
449	2	1	1418	1388	Benign
290	4	1	0	0	Benign
412	2	1	1418	1388	Benign
177	1	1	1388	1388	Benign
209	2	1	2776	1388	Benign
78	0	2	0	2896	Benign
75	1	1	982	1430	Zero-day
54	2	3	0	4076	Zero-day
165	1	1	1441	1441	Zero-day
199	2	1	2800	1400	Zero-day
212	3	1	4290	1430	Zero-day
40	10	1	320	32	Zero-day
60431	5	7	5680	1097	Zero-day
293	0	5	0	168	Zero-day

Figure 5.3: IoT Intrusion-2020 - benign and zero-day sample

The features of the IoT Intrusion-2020 data set in Table 5.1 are again presented in Tables C.1, C.2 and C.3 in Appendix C.1, with feature descriptions found in the first row. Information is provided regarding data type - e.g., numeric and non-numeric, minimum, and maximum values for each feature in the IoT Intrusion-2020 data set. This information will be key when feature selection is implemented.

Flow_ID	Src_IP	Src_Port	Dst_IP	Dst_Port	Protocol	Flow_Byts	Flow_Pkts	Flow_IAT	Flow_IAT	Flow_IAT	Flow_IAT	Fwd_IAT	Fwd_IAT	Bwd_IAT	Bwd_IAT	Bwd_IAT	Bwd_IAT	Bwd_IAT	Fwd_Pkts	Bwd_Pkts	Pkt_Len_M	Pkt_Len_M	Pkt_Len_M	Pkt_Len_S	Pkt_Len_V	Label	Sub_Cat
192.168.0.	192.168.0.	10000	192.168.0.	10101	17	32160000	26666.67	75	0	75	75	0	0	0	0	0	0	0	13333.33	13333.33	982	1430	1280.667	258.6529	66901.33	Anomaly	Mirai-Ackflooding
192.168.0.	222.160.17	2179	192.168.0.	554	6	0	564.9718	2655	2261.327	4254	1056	0	0	5310	5310	0	5310	5310	188.3239	376.6478	0	0	0	0	0	Anomaly	DoS-Synflooding
192.168.0.	192.168.0.	52727	192.168.0.	9020	6	19900709	21276.6	70.5	0.707107	71	70	0	0	141	70.5	0.707107	71	70	0	21276.6	30	1388	1048.5	679	461041	Anomaly	Scan Port OS
192.168.0.	192.168.0.	52964	192.168.0.	9020	6	18384106	13245.03	151	0	151	151	0	0	151	151	0	151	151	0	13245.03	1388	1388	1388	0	0	Anomaly	Mirai-Hostbruteforceg
192.168.0.	192.168.0.	36763	239.255.25	1900	17	8535948	19607.84	76.5	0.707107	77	76	76	76	0	0	0	0	0	13071.9	6535.948	420	452	431.5	15.17674	230.3333	Anomaly	Mirai-Hostbruteforceg
192.168.0.	192.168.0.	41980	101.79.244	443	6	0	19108.28	78.5	6.363961	83	74	74	74	0	0	0	0	0	12738.85	6369.427	0	0	0	0	0	Anomaly	Mirai-Hostbruteforceg
192.168.0.	192.168.0.	60175	210.89.164	8899	17	4834532	151079.1	6.95	1.669384	10	4	132	6.947368	0	0	0	0	0	143884.9	7194.245	32	32	32	0	0	Anomaly	Mirai-UDP Flooding
192.168.0.	192.168.0.	41467	58.225.75.	443	6	25857143	17857.14	112	0	112	112	0	0	112	112	0	112	112	0	17857.14	1448	1448	1448	0	0	Anomaly	Scan Port OS
192.168.0.	192.168.0.	60132	210.89.164	8899	17	744186	23255.81	86	0	86	86	0	0	0	0	0	0	0	11627.91	11627.91	32	32	32	0	0	Anomaly	Mirai-UDP Flooding
192.168.0.	111.149.16	7953	192.168.0.	554	6	0	294.1609	6799	0	6799	6799	0	0	6799	6799	0	6799	6799	0	294.1609	0	0	0	0	0	Anomaly	DoS-Synflooding
192.168.0.	192.168.0.	10102	40.100.49.	443	6	75481481	92592.59	13.5	4.795832	20	9	40	40	43	21.5	2.12132	23	20	37037.04	55555.56	0	1460	922.6667	724.3277	524650.7	Anomaly	Mirai-Hostbruteforceg
192.168.0.	104.118.13	443	192.168.0.	43238	6	17466667	12121.21	165	0	165	165	0	0	0	0	0	0	0	6060.606	6060.606	1441	1441	1441	0	0	Anomaly	Mirai-UDP Flooding
192.168.0.	192.168.0.	53604	52.219.36.	443	6	21105528	15075.38	99.5	33.23402	123	76	76	76	0	0	0	0	0	10050.25	5025.126	1400	1400	1400	0	0	Anomaly	Mirai-UDP Flooding
192.168.0.	192.168.0.	10000	192.168.0.	10101	17	26981132	18867.92	70.66667	0.57735	71	70	141	70.5	0	0	0	0	0	14150.94	4716.981	1430	1430	1430	0	0	Anomaly	Mirai-HTTP Flooding
192.168.0.	192.168.0.	60079	210.89.164	8899	17	8800000	275000	4	1.563472	7	1	37	4.111111	0	0	0	0	0	250000	25000	32	32	32	0	0	Anomaly	Mirai-UDP Flooding
192.168.0.	192.168.0.	9020	192.168.0.	44144	6	112144.4	198.5736	5493.727	12649.92	43394	121	56804	14201	60431	10071.83	18541.28	47389	371	82.73899	115.8346	0	1388	605.6923	589.7424	347796.1	Anomaly	DoS-Synflooding
192.168.0.	192.168.0.	56361	192.168.0.	10101	17	573378.8	17064.85	73.25	0.5	74	73	0	0	293	73.25	0.5	74	73	0	17064.85	32	40	34.66667	4.131182	17.06667	Anomaly	Mirai-HTTP Flooding
192.168.0.	192.168.0.	9020	192.168.0.	49784	6	11816667	16666.67	120	0	120	120	0	0	0	0	0	0	0	8333.333	8333.333	30	1388	935.3333	784.0417	614721.3	Normal	Normal
192.168.0.	192.168.0.	56361	192.168.0.	10101	17	467532.5	12987.01	154	0	154	154	0	0	154	154	0	154	154	0	12987.01	32	40	37.33333	4.618802	21.33333	Anomaly	Mirai-Ackflooding
192.168.0.	192.168.0.	60183	210.89.164	8899	17	359550.6	11235.96	133.5	185.9691	265	2	265	265	0	0	0	0	0	7490.637	3745.318	32	32	32	0	0	Anomaly	Mirai-UDP Flooding
192.168.0.	192.168.0.	10000	192.168.0.	10101	17	22698413	15873.02	84	25.11971	113	69	183	91.5	0	0	0	0	0	11904.76	3968.254	1430	1430	1430	0	0	Anomaly	Mirai-Ackflooding
192.168.0.	104.74.213	443	192.168.0.	51875	6	24648649	27027.03	74	0	74	74	0	0	0	0	0	0	0	13513.51	13513.51	376	1448	1090.667	618.9195	383061.3	Anomaly	Mirai-Hostbruteforceg
192.168.0.	192.168.0.	9020	192.168.0.	52717	6	0	16260.16	123	0	123	123	0	0	123	123	0	123	123	0	16260.16	0	0	0	0	0	Anomaly	Scan Port OS
192.168.0.	192.168.0.	10000	192.168.0.	10101	17	7396985	10050.25	199	0	199	199	0	0	0	0	0	0	0	5025.126	5025.126	42	1430	967.3333	801.3622	642181.3	Anomaly	Mirai-Ackflooding
192.168.0.	192.168.0.	56260	124.156.24	7100	6	19916084	13986.01	143	0	143	143	0	0	143	143	0	143	143	0	13986.01	1424	1424	1424	0	0	Anomaly	Mirai-Hostbruteforceg
192.168.0.	192.168.0.	56361	192.168.0.	10101	17	22113402	15463.92	97	33.94113	121	73	73	73	0	0	0	0	0	10309.28	5154.639	1430	1430	1430	0	0	Anomaly	Mirai-Ackflooding
192.168.0.	192.168.0.	56361	192.168.0.	10101	17	596491.2	17543.86	76	1	77	75	0	0	228	76	1	77	75	0	17543.86	32	40	35.2	4.38178	19.2	Anomaly	Mirai-Ackflooding
192.168.0.	192.168.0.	56361	192.168.0.	10101	17	20483516	14652.01	91	32.04684	128	72	200	100	0	0	0	0	0	10989.01	3663.004	1302	1430	1404.4	57.24334	3276.8	Anomaly	Mirai-UDP Flooding
192.168.0.	192.168.0.	52739	192.168.0.	9020	6	0	16129.03	124	0	124	124	0	0	0	0	0	0	0	8064.516	8064.516	0	0	0	0	0	Anomaly	Scan Hostport
180.182.6	192.168.0.	43987	180.182.6	443	6	28392157	19607.84	76.5	0.707107	77	76	0	0	153	76.5	0.707107	77	76	0	19607.84	1448	1448	1448	0	0	Anomaly	Mirai-Hostbruteforceg
192.168.0.	192.168.0.	52727	192.168.0.	9020	6	0	27397.26	73	0	73	73	0	0	0	0	0	0	0	13698.63	13698.63	0	0	0	0	0	Anomaly	Scan Port OS
192.168.0.	192.168.0.	10000	192.168.0.	10101	17	17565657	15151.52	99	36.76955	125	73	73	73	0	0	0	0	0	10101.01	5050.505	618	1430	1227	406	164836	Anomaly	Mirai-Ackflooding
150.109.16	150.109.16	80	192.168.0.	56369	6	18782609	13043.48	115	57.98276	156	74	156	156	0	0	0	0	0	8695.652	4347.826	1440	1440	1440	0	0	Anomaly	Mirai-Ackflooding
192.168.0.	192.168.0.	9020	192.168.0.	52919	6	9516779	13422.82	149	0	149	149	0	0	0	0	0	0	0	6711.409	6711.409	30	1388	935.3333	784.0417	614721.3	Anomaly	Mirai-Hostbruteforceg
192.168.0.	192.168.0.	51252	222.239.24	443	6	12323404	8510.638	235	0	235	235	0	0	235	235	0	235	235	0	8510.638	1448	1448	1448	0	0	Anomaly	Mirai-HTTP Flooding
192.168.0.	192.168.0.	10000	192.168.0.	10101	17	259740.3	12987.01	154	0	154	154	0	0	0	0	0	0	0	6493.506	6493.506	20	20	20	0	0	Anomaly	Mirai-UDP Flooding

Figure 5.4: IoT Intrusion-2020 sample data set

Figure 5.4 provides an extract of the IoT Intrusion-2020 data set. As depicted in this figure, the majority of features such as flow packets have numeric values. Given that most of these features are numeric, the implementation of a feature selection method should be straightforward, as feature selection methods require data to be in numerical form [244, 206].

The UNSW-NB15 network intrusion data set is introduced next.

5.3.1.2 UNSW-NB15 data set

Moustafa et al. [2] developed a network intrusion data set called UNSW-NB15 that contains bot related attacks [263, 264]. This data set was developed in 2015, based on various botnet synthetic attacks such as DoS and worms. The authors of this data set used the IXIA PerfectStorm tool [2] to generate network traffic data. The IXIA PerfectStorm tool has more than 49 features, which include source port address, see Figure 5.6.

Network traffic	**Description**	**Number of instances**
Benign	Normal network traffic.	56000
DoS	"The attacker attempts to deny an authorised person access to a network or device."	12264
Generic collision	"The attacker attempts to collide block-cyphers using the hash function."	40000
Worm	"The attacker uses malware that can replicate itself rapidly and spread across devices in a network."	130
Shell-code	"This is a malware code used by an attacker that attempts to control a compromised device."	1133
Recon-naissance	"The attacker attempts to illegally collect network information so as to bypass the security controls."	10491
Fuzzing	"The attacker attempts to overload a network by feeding it massive random data."	18184
Analysis	"The attacker analyses a network structure to determine network ports and nodes that can be exploited."	2000
Back-door	"The attacker attempts to bypass a customary network security control."	1746
Exploit	"This is a malware code that exploits a vulnerability in a network that intends to cause the network to malfunction."	33393

Table 5.2: The UNSW-NB15 network traffic types and descriptions adopted from [2]

The benign network traffic is labelled as "normal", while malicious network traffic is labelled as "anomaly" [2]. For the purposes of this thesis, the author of this thesis manually created a zero-day network intrusion attack type by combining anomalous network traffic, i.e., zero-day attack = DoS + Generic collision + Worm + ShellCode + Reconnaissance + Fuzzing + Analysis + Backdoor + Exploit across the same features. Figure 5.5 below illustrates a sample of the zero-day attack created by the author of this thesis.

spkts	dpkts	sbytes	dbytes	rate	Label
6	3	258	172	74.08749	Benign
14	38	734	42014	78.47337	Benign
8	16	364	13186	14.17016	Benign
12	12	628	770	13.67711	Benign
10	3	534	268	33.37383	Benign
10	3	534	268	39.41798	Benign
10	8	534	354	26.68303	Benign
10	8	534	354	32.59303	Benign
10	8	534	354	31.31303	Benign
10	3	534	268	57.98514	Benign
12	3	4142	268	55.76458	Benign
62	28	56329	2212	42.52097	Benign
10	3	534	268	35.97536	Benign
10	8	564	354	17.06449	Benign
10	6	200	0	111111.1	Zero-day
10	6	200	0	111111.1	Zero-day
10	6	200	0	111111.1	Zero-day
10	6	200	0	333333.3	Zero-day
10	6	200	0	125000	Zero-day
10	6	200	0	200000	Zero-day
10	6	200	0	125000	Zero-day
10	6	200	0	200000	Zero-day

Figure 5.5: UNSW-NB15 - benign and zero-day sample

Appendix C.2 contains a full description of the UNSW-NB15 features.

dur	proto	service	state	spkts	dpkts	sbytes	dbytes	rate	sttl	dttl	sload	dload	sloss	dloss	sinpkt	dinpkt	sjit	djit	swin	stcpb	dtcpb	dwin	tcprtt	synack	ackdat	smean	attack_cat	label
29.19551	tcp	-	REQ	4	0	180	0	0.102756	254	0	36.99199	0	3	0	9731.837	0	13762.88	0	255	0	0	0	0	0	0	45	Normal	0
30.79278	tcp	-	REQ	4	0	180	0	0.097425	254	0	35.07316	0	3	0	10264.26	0	14515.85	0	255	0	0	0	0	0	0	45	Normal	0
30.79278	tcp	-	REQ	4	0	180	0	0.097425	254	0	35.07316	0	3	0	10264.26	0	14515.85	0	255	0	0	0	0	0	0	45	Normal	0
27.41393	tcp	-	REQ	4	0	180	0	0.109433	254	0	39.39603	0	3	0	9137.977	0	12923.05	0	255	0	0	0	0	0	0	45	Normal	0
31.54256	tcp	-	REQ	4	0	180	0	0.09511	254	0	34.23945	0	3	0	10514.19	0	14869.29	0	255	0	0	0	0	0	0	45	Normal	0
0.524961	tcp	-	FIN	10	6	534	268	28.57355	254	252	7330.068	3413.587	2	1	58.22633	79.2218	2984.402	121.2344	255	2.57E+09	3.45E+09	255	0.197129	0.128843	0.068286	53	Normal	0
0.000003	udp	-	INT	2	0	104	0	333333.3	254	0	1.39E+08	0	0	0	0.003	0	0	0	0	0	0	0	0	0	0	52	Normal	0
29.19551	tcp	-	REQ	4	0	180	0	0.102756	254	0	36.99199	0	3	0	9731.837	0	13762.88	0	255	0	0	0	0	0	0	45	Normal	0
27.41393	tcp	-	REQ	4	0	180	0	0.109433	254	0	39.39603	0	3	0	9137.977	0	12923.05	0	255	0	0	0	0	0	0	45	Normal	0
0.000005	igmp	-	INT	2	0	90	0	200000	254	0	72000000	0	0	0	0.005	0	0	0	0	0	0	0	0	0	0	45	Normal	0
0.000005	igmp	-	INT	2	0	90	0	200000	254	0	72000000	0	0	0	0.005	0	0	0	0	0	0	0	0	0	0	45	Normal	0
1.129944	tcp	-	FIN	10	8	534	354	15.04499	254	252	3405.479	2194.799	2	1	122.617	150.6929	6633.796	259.9588	255	1.37E+09	1.19E+09	255	0.248763	0.075085	0.173678	53	Normal	0
1.964566	tcp	-	FIN	10	8	2516	354	8.653311	254	252	9223.411	1262.365	2	1	218.1742	270.6407	13125.1	469.482	255	1.59E+09	2.48E+09	255	0.159326	0.07008	0.089246	252	Normal	0
2.349783	tcp	http	FIN	12	10	888	1370	8.936996	62	252	2771.32	4197.834	2	2	213.6166	241.403	12385.52	352.7233	255	1.79E+09	2.63E+09	255	0.335828	0.176119	0.159709	74	Normal	0
0.000007	udp	-	INT	2	0	104	0	142857.1	254	0	59428572	0	0	0	0.007	0	0	0	0	0	0	0	0	0	0	52	Normal	0
0.921987	ospf	-	INT	20	0	1280	0	20.60767	254	0	10551.13	0	0	0	48.52563	0	52.25381	0	0	0	0	0	0	0	0	64	Reconnais	1
0.921987	ospf	-	INT	20	0	1280	0	20.60767	254	0	10551.13	0	0	0	48.52563	0	52.25381	0	0	0	0	0	0	0	0	64	Reconnais	1
0.921987	ospf	-	INT	20	0	1280	0	20.60767	254	0	10551.13	0	0	0	48.52563	0	52.25381	0	0	0	0	0	0	0	0	64	Backdoor	1
0.921987	ospf	-	INT	20	0	1280	0	20.60767	254	0	10551.13	0	0	0	48.52563	0	52.25381	0	0	0	0	0	0	0	0	64	DoS	1
0.000009	sctp	-	INT	2	0	104	0	111111.1	254	0	46222220	0	0	0	0.009	0	0	0	0	0	0	0	0	0	0	52	Exploits	1
0.000009	gre	-	INT	2	0	156	0	111111.1	254	0	69333328	0	0	0	0.009	0	0	0	0	0	0	0	0	0	0	78	Analysis	1
0.000009	gre	-	INT	2	0	156	0	111111.1	254	0	69333328	0	0	0	0.009	0	0	0	0	0	0	0	0	0	0	78	Exploits	1
0.000009	gre	-	INT	2	0	156	0	111111.1	254	0	69333328	0	0	0	0.009	0	0	0	0	0	0	0	0	0	0	78	Exploits	1
0.000009	gre	-	INT	2	0	156	0	111111.1	254	0	69333328	0	0	0	0.009	0	0	0	0	0	0	0	0	0	0	78	Exploits	1
0.000016	udp	-	INT	2	0	196	0	62500	254	0	49000000	0	0	0	0.016	0	0	0	0	0	0	0	0	0	0	98	Exploits	1
0.224524	tcp	http	FIN	10	8	830	960	75.71573	62	252	26616.31	29929.98	2	2	24.94711	30.40743	1344.946	54.34261	255	3.31E+09	3.28E+08	255	0.055643	0.00698	0.048663	83	Exploits	1
0.409782	tcp	http	FIN	10	8	852	1336	41.48547	62	252	14973.82	22821.89	2	2	45.53133	46.37415	2261.112	78.61516	255	1.11E+08	1.03E+09	255	0.123943	0.056063	0.06788	85	Exploits	1
0.258515	tcp	http	FIN	10	6	1064	268	58.02371	254	252	29646.25	6931.899	2	1	28.72389	43.3266	1393.321	76.35581	255	1.11E+09	46488383	255	0.05861	0.011825	0.046785	106	Exploits	1
0.674665	tcp	-	FIN	10	8	564	354	25.19769	254	252	6023.73	3675.898	2	1	74.96278	90.62814	5394.127	130.207	255	2.49E+09	4.07E+09	255	0.07321	0.039567	0.033643	56	Reconnais	1
5.600391	tcp	http	FIN	12	12	886	5236	4.106856	62	252	1161.348	6856.664	2	3	509.1265	505.8742	38893.67	36169.43	255	3.33E+09	2.68E+09	255	0.060118	0.010067	0.050051	74	Exploits	1
0.285988	tcp	-	FIN	10	8	1196	992	59.44305	62	252	30127.14	24280.74	2	2	31.77644	37.35	1638.963	58.91657	255	3.63E+09	1.49E+09	255	0.069484	0.012339	0.057145	120	Exploits	1
0.000003	udp	-	INT	2	0	168	0	333333.3	254	0	2.24E+08	0	0	0	0.003	0	0	0	0	0	0	0	0	0	0	84	Reconnais	1
0.663844	tcp	-	FIN	10	8	564	354	25.60843	254	252	6121.92	3735.818	2	1	73.76044	87.52443	5551.276	137.1557	255	7.02E+08	1.4E+08	255	0.104112	0.038465	0.065647	56	Reconnais	1
0.217312	tcp	http	FIN	10	6	1148	268	69.02518	254	252	38065.09	8246.208	2	1	24.14578	41.3548	1254.08	61.81749	255	1.35E+09	3.18E+09	255	0.024186	0.009806	0.01438	115	Reconnais	1
7.629429	tcp	http	FIN	202	1294	9082	1747118	195.9518	62	252	9476.987	1830562	2	645	37.95736	5.882331	1929.953	745.6064	255	9.46E+08	4.12E+09	255	0.052951	0.028738	0.024213	45	Exploits	1
0.502392	tcp	http	FIN	10	8	938	354	33.83812	254	252	13455.63	4936.384	2	1	53.853	62.081	3382.951	125.0175	255	2.13E+08	1.34E+09	255	0.10843	0.067821	0.040609	94	Exploits	1

Figure 5.6: UNSW-NB15 sample data set

Figure 5.6 provides an extract of the UNSW-NB15 data set. Once more, most of these features have numerical values such as source to destination packets. Features such as transaction protocol ("proto"), which are non-numerical, are not considered in this thesis, as they do not provide much insights in terms of benign and malicious network traffic [178].

The CICDDOS2019 network intrusion data set is introduced next.

5.3.1.3 CICDDOS2019 data set

In 2019, the Canadian Institute for Cybersecurity [9] developed a network intrusion data set called CICDDOS2019 based on various synthetic DDoS botnet attacks, as well as on real benign network traffic. They used the CICFlowMeter tool [233] to generate network traffic data. Ullah et al. [8], authors of the IoT Intrusion-2020 data set, also used the CICFlowMeter tool to generate network traffic data. Therefore, features of the CICDDOS2019 intrusion data set are the same as for the IoT Intrusion-2020 data set, although the network traffic data is different.

Network traffic	**Description**	**Number of instances**
Benign	Normal network traffic.	26719
DDoS_DNS	"The attacker uses DNS to execute DDoS."	1046567
DDoS_LDAP	"The attacker uses LDAP to execute DDoS."	1047795
DDoS_MSSQL	"The attacker uses MSSQL to execute DDoS."	1047561
DDoS_NetBIOS	"The attacker uses NetBIOS to execute DDoS."	1047706
DDoS_NTP	"The attacker uses NTP to execute DDoS."	1035009
DDoS_SNMP	"The attacker uses SNMP to execute DDoS."	1047663
DDoS_SSDP	"The attacker uses SSDP to execute DDoS."	1048271
DDoS_UDP	"The attacker uses UDP to execute DDoS."	1047441
DDoS_SYN	"The attacker uses SYN to execute DDoS."	1048228
DDoS_TFTP	"The attacker uses TFTP to execute DDoS."	1047402
DDoS_UDPLag / Web	"The attacker uses UDPLag or Web to execute DDoS."	366900

Table 5.3: The CICDDOS2019 network traffic types and descriptions adopted from [9]

The benign network traffic is labelled as “normal”, while malicious network traffic are labelled as “anomaly” [9]. For the purposes of this research, the author of this thesis created a zero-day network intrusion attack manually by combining anomalous network traffic across the same features. Figure 5.7 below illustrates a sample of the zero-day attack created by the author of this thesis.

Flow Duration	Total Fwd Packets	Total Backward Packets	Total Length of Fwd Packets	Total Length of Bwd Packets	Label
40694755	5	0	1500	0	Benign
113871580	52	0	0	0	Benign
112936309	39	0	0	0	Benign
24944828	8	6	37	0	Benign
20779	2	2	86	366	Benign
118074966	36	48	2474	67854	Benign
20568	2	2	86	118	Benign
65675	2	2	86	118	Benign
20684	2	2	86	198	Benign
90699	4	2	62	70	Benign
117101485	12	0	0	0	Benign
1	2	0	31	0	Benign
4517500	9	6	1180	2146	Zero-day
5002555	8	7	1338	2146	Zero-day
148	1	1	0	0	Zero-day
100	1	2	0	0	Zero-day
179	1	1	0	0	Zero-day
109	1	2	0	0	Zero-day
5318128	13	8	2298	4290	Zero-day

Figure 5.7: CICDDOS2019 - benign and zero-day sample

Appendix C.3 contains a full description of the CICDDOS2019 features.

Flow ID	Source IP	Source Po	Destinati	Destinati	Timestam	Total Leng	Fwd Pack	Fwd Pack	Fwd Pack	Fwd Pack	Flow Byte	Flow Pack	Flow IAT I	Flow IAT S	Flow IAT I	Fwd IAT To	Fwd IAT M	Fwd IAT S	Fwd IAT M	Fwd Packe	Min Packe	Max Pack	Packet Le	Packet Le	Packet Le	Average F	Avg Fwd S	Label
172.16.0.5	172.16.0.5	58445	192.168.5(	4463	04:45.9	766	383	383	383	0	7.66E+08	2000000	1	0	1	1	1	0	1	2000000	383	383	383	0	0	574.5	383	UDP-lag
172.16.0.5	172.16.0.5	36908	192.168.5(	9914	04:45.9	778	389	389	389	0	7.78E+08	2000000	1	0	1	1	1	0	1	2000000	389	389	389	0	0	583.5	389	UDP-lag
172.16.0.5	172.16.0.5	41727	192.168.5(	32361	04:45.9	750	375	375	375	0	3.75E+08	1000000	2	0	2	2	2	0	2	1000000	375	375	375	0	0	562.5	375	UDP-lag
172.16.0.5	172.16.0.5	55447	192.168.5(	5691	04:45.9	738	369	369	369	0	3.69E+08	1000000	2	0	2	2	2	0	2	1000000	369	369	369	0	0	553.5	369	UDP-lag
172.16.0.5	172.16.0.5	58794	192.168.5(	56335	04:45.9	750	375	375	375	0	7.50E+08	2000000	1	0	1	1	1	0	1	2000000	375	375	375	0	0	562.5	375	UDP-lag
172.16.0.5	172.16.0.5	48002	192.168.5(	57615	04:45.9	1438	389	330	359.5	34.06367	13512.24	37.58621	35474	61441.04	106420	106422	35474	61441.04	106420	37.58621	330	389	353.6	32.31563	1044.3	442	359.5	UDP-lag
172.16.0.5	172.16.0.5	51718	192.168.5(	34159	04:45.9	778	389	389	389	0	7.78E+08	2000000	1	0	1	1	1	0	1	2000000	389	389	389	0	0	583.5	389	UDP-lag
172.16.0.5	172.16.0.5	55448	192.168.5(	12269	04:45.9	738	369	369	369	0	7.38E+08	2000000	1	0	1	1	1	0	1	2000000	369	369	369	0	0	553.5	369	UDP-lag
172.16.0.5	172.16.0.5	58793	192.168.5(	36639	04:45.9	750	375	375	375	0	7.50E+08	2000000	1	0	1	1	1	0	1	2000000	375	375	375	0	0	562.5	375	UDP-lag
172.16.0.5	172.16.0.5	35276	192.168.5(	36368	04:45.9	7242	401	321	362.1	30.12762	7320.49	20.21676	52067.26	56392.75	112381	989278	52067.26	56392.75	112381	20.21676	321	401	360.1429	30.70389	942.7286	378.15	362.1	UDP-lag
172.16.0.5	172.16.0.5	60389	192.168.5(	31463	04:45.9	2088	393	321	348	35.08846	9555.189	27.45744	43704	59842.94	109442	218520	43704	59842.94	109442	27.45744	321	393	344.1429	33.6176	1130.143	401.5	348	UDP-lag
172.16.0.5	172.16.0.5	39381	192.168.5(	53349	04:45.9	1446	393	330	361.5	36.37307	13135.54	36.33622	36694.33	63513.15	110033	110083	36694.33	63513.15	110033	36.33622	330	393	355.2	34.50652	1190.7	444	361.5	UDP-lag
172.16.0.5	172.16.0.5	36067	192.168.5(	46692	04:45.9	766	383	383	383	0	7.66E+08	2000000	1	0	1	1	1	0	1	2000000	383	383	383	0	0	574.5	383	UDP-lag
172.16.0.5	172.16.0.5	43351	192.168.5(	13670	04:45.9	1446	393	330	361.5	36.37307	13076.39	36.17258	36860.33	63841.37	110578	110581	36860.33	63841.37	110578	36.17258	330	393	355.2	34.50652	1190.7	444	361.5	UDP-lag
172.16.0.5	172.16.0.5	37756	192.168.5(	52783	04:45.9	1398	369	330	349.5	22.51666	12958.72	37.07789	35960.33	62283.39	107879	107881	35960.33	62283.39	107879	37.07789	330	369	345.6	21.36118	456.3	432	349.5	UDP-lag
172.16.0.5	172.16.0.5	51556	192.168.5(	38622	04:45.9	786	393	393	393	0	7.86E+08	2000000	1	0	1	1	1	0	1	2000000	393	393	393	0	0	589.5	393	UDP-lag
172.16.0.5	172.16.0.5	50582	192.168.5(	33248	04:45.9	802	401	401	401	0	8.02E+08	2000000	1	0	1	1	1	0	1	2000000	401	401	401	0	0	601.5	401	UDP-lag
172.16.0.5	172.16.0.5	37710	192.168.5(	5225	04:45.9	1438	389	330	359.5	34.06367	13365.55	37.17818	35863.33	62115.38	107588	107590	35863.33	62115.38	107588	37.17818	330	389	353.6	32.31563	1044.3	442	359.5	UDP-lag
172.16.0.5	172.16.0.5	45262	192.168.5(	65248	04:45.9	738	369	369	369	0	3.69E+08	1000000	2	0	2	2	2	0	2	1000000	369	369	369	0	0	553.5	369	UDP-lag
172.16.0.5	172.16.0.5	39156	192.168.5(	20460	04:45.9	1446	393	330	361.5	36.37307	13155.98	36.39275	36637.33	63455.99	109910	109912	36637.33	63455.99	109910	36.39275	330	393	355.2	34.50652	1190.7	444	361.5	UDP-lag
172.16.0.5	172.16.0.5	58260	192.168.5(	51629	04:45.9	778	389	389	389	0	7.78E+08	2000000	1	0	1	1	1	0	1	2000000	389	389	389	0	0	583.5	389	UDP-lag
172.16.0.5	172.16.0.5	48637	192.168.5(	32341	04:45.9	1438	389	330	359.5	34.06367	13435.86	37.37375	35675.67	61790.34	107025	107027	35675.67	61790.34	107025	37.37375	330	389	353.6	32.31563	1044.3	442	359.5	UDP-lag
172.16.0.5	172.16.0.5	41929	192.168.5(	35580	04:45.9	2088	393	321	348	35.08846	9700.484	27.87495	43049.4	58964.5	109685	215247	43049.4	58964.5	109685	27.87495	321	393	344.1429	33.6176	1130.143	401.5	348	UDP-lag
172.16.0.5	172.16.0.5	45263	192.168.5(	28912	04:45.9	738	369	369	369	0	7.38E+08	2000000	1	0	1	1	1	0	1	2000000	369	369	369	0	0	553.5	369	UDP-lag
172.16.0.5	172.16.0.5	48004	192.168.5(	33924	04:45.9	1438	389	330	359.5	34.06367	13642.62	37.94886	35135	60853.01	105402	105405	35135	60853.01	105402	37.94886	330	389	353.6	32.31563	1044.3	442	359.5	UDP-lag
172.16.0.5	172.16.0.5	44969	192.168.5(	17617	04:45.9	786	393	393	393	0	1.57E+07	40000	50	0	50	50	50	0	50	40000	393	393	393	0	0	589.5	393	UDP-lag
172.16.0.5	172.16.0.5	43992	192.168.5(	20093	04:45.9	802	401	401	401	0	8.02E+08	2000000	1	0	1	1	1	0	1	2000000	401	401	401	0	0	601.5	401	UDP-lag
172.16.0.5	172.16.0.5	53127	192.168.5(	60972	04:45.9	750	375	375	375	0	7.50E+08	2000000	1	0	1	1	1	0	1	2000000	375	375	375	0	0	562.5	375	UDP-lag
172.16.0.5	172.16.0.5	36909	192.168.5(	30240	04:45.9	778	389	389	389	0	7.78E+08	2000000	1	0	1	1	1	0	1	2000000	389	389	389	0	0	583.5	389	UDP-lag
172.16.0.5	172.16.0.5	58446	192.168.5(	43404	04:45.9	766	383	383	383	0	7.66E+08	2000000	1	0	1	1	1	0	1	2000000	383	383	383	0	0	574.5	383	UDP-lag
172.16.0.5	172.16.0.5	58406	192.168.5(	4693	04:45.9	2088	393	321	348	35.08846	9722.753	27.93894	42950.8	58811.62	109631	214754	42950.8	58811.62	109631	27.93894	321	393	344.1429	33.6176	1130.143	401.5	348	UDP-lag
172.16.0.5	172.16.0.5	37291	192.168.5(	15736	04:45.9	1398	369	330	349.5	22.51666	13219.48	37.82399	35251	61054.79	105751	105753	35251	61054.79	105751	37.82399	330	369	345.6	21.36118	456.3	432	349.5	UDP-lag
172.16.0.5	172.16.0.5	40676	192.168.5(	58590	04:45.9	778	389	389	389	0	7.78E+08	2000000	1	0	1	1	1	0	1	2000000	389	389	389	0	0	583.5	389	UDP-lag
172.16.0.5	172.16.0.5	41830	192.168.5(	19829	04:45.9	750	375	375	375	0	7.50E+08	2000000	1	0	1	1	1	0	1	2000000	375	375	375	0	0	562.5	375	UDP-lag
172.16.0.5	172.16.0.5	45473	192.168.5(	58305	04:45.9	2088	393	321	348	35.08846	9613.083	27.6238	43440.8	59494.26	110312	217204	43440.8	59494.26	110312	27.6238	321	393	344.1429	33.6176	1130.143	401.5	348	UDP-lag
172.16.0.5	172.16.0.5	58262	192.168.5(	21314	04:45.9	778	389	389	389	0	7.78E+08	2000000	1	0	1	1	1	0	1	2000000	389	389	389	0	0	583.5	389	UDP-lag

Figure 5.8: CICDDOS2019 sample data set

Figure 5.8 provides an extract of the CICDDOS2019 data set. Once more, the majority of features in this data set have numeric values.

The CIRACICDOHBRW2020 data set is introduced next.

5.3.1.4 CIRACICDOHBRW2020 data set

The Canadian Institute for Cybersecurity [10] developed a network intrusion data set called CIRACICDOHBRW2020 based on synthetic attacks. This data set was developed in 2020 based on various synthetic domain names over HTTPS (DoH), non-DoH, benign, and malicious network traffic. The authors of this data set used the DoHMeter [265, 266], an open-access tool, to generate network traffic data. The DoHMeter has about 35 features, including source and destination IP addresses. Domain name system (DNS) is one protocol known to be abused by botmasters to launch cybercrimes and bypass security controls [267, 268, 269]. In this instance, a botmaster uses Domain generative algorithms (DGAs) to generate multiple domain names with only a few that are registered and uses them as C&C servers to communicate with its bots, i.e., botnets [267, 268]. The CIRACICDOHBRW2020 data set was chosen in this thesis because it is the most recent data set containing modern DoH attacks known to be exploited by botnets [267, 268, 269].

Network traffic	Description	Number of instances
Non-DoH	"Non-DoH network traffic."	897493
DoH	"DoH network traffic."	269643
Benign-DoH	"Benign network traffic."	19807
Malicious-DoH	"Malicious network traffic."	249836

Table 5.4: The CIRACICDOHBRW2020 network traffic types and descriptions adopted from [10]

In this thesis, the author of this thesis created a zero-day network attack by combining DoH and malicious DoH network traffic [260], i.e., zero-day attack = DoH + Malicious-DoH across the same features. Figure 5.9 below illustrates a sample of the zero-day attack created by the author of this thesis.

Duration	FlowBytesSent	FlowSentRate	FlowBytesReceived	FlowReceivedRate	Label
95.08155	62311	655.342703184792422925373 0087	65358	687.388878283957297709177 0170	Benign
122.309318	93828	767.136973161766791962653 2461	101232	827.672017597220188898445 1700	Benign
120.958413	38784	320.639127432996330730628 8815	38236	316.108644712459975810033 1558	Benign
110.50108	61993	561.017141189932261295545 7087	69757	631.278897907604160972906 3282	Benign
54.229891	83641	1542.341289234750628578619 124	76804	1416.266907119544090545931 579	Benign
145.460721	54084	371.811714036533615146868 4113	63843	438.901990593048139779260 4094	Benign
0.15541	1718	11054.629689209188597902322 89	7411	47686.764043497844411556527 89	Benign
0.027001	55	2036.961594015036480130365542	66	2444.353912818043776156438650	Benign
44.649554	32285	723.075531728715588066120 4365	36193	810.601601977927931822118 5367	Benign
0.046455	55	1183.941448713809062533634700	66	1420.729738456570875040361640	Benign
44.890412	163	3.631064914262760609102897073	357	7.952700456391445015029044510	Benign
91.338058	8090	88.572060509541378687950645 94	7805	85.451783965014890069153867 93	Benign
121	46446	384.705256724910205414765 8319	85465	707.893785600362802087864 6563	Zero-day
120	470548	3919.510349246390863629605622	227661	1896.341384130381152198669733	Zero-day
120	747083	6223.907583557839811884589949	355181	2958.993471188150583301963159	Zero-day
120	745382	6211.268008904043541456890342	354606	2954.931704234107161243311559	Zero-day
120	749888	6247.732775719050649353029795	356531	2970.457473995968527386069741	Zero-day
110	687007	6246.041486530362398362393707	326602	2969.357869110197450724598890	Zero-day
64	374454	5865.568342829620194369810038	182045	2851.611650484220246770102785	Zero-day
120	270012	2249.001456496895287356096853	552995	4606.041807162276526345069031	Zero-day

Figure 5.9: CIRACICDOHBRW2020 - benign and zero-day sample

Appendix C.4 contains a full description of the CIRACICDOHBRW2020 features.

SourceIP	Destinatic	SourcePo	Destinatic	FlowBytes	FlowSentI	FlowBytes	FlowRece	PacketLen	PacketLen	PacketLen	PacketLen	PacketLen	PacketLen	PacketTim	PacketTim	PacketTim	PacketTim	PacketTim	PacketTim	PacketTim	Response	Response	Response	Response	Response	Response	Label
192.168.2(	176.103.13	50749	443	62311	655.3427	65358	687.3889	7474.677	86.45621	135.6738	1.168467	0.944683	0.637236	670.5858	25.89567	45.06528	48.81129	1.49506	-0.43397	1.682529	0.574626	0.001053	0.032457	0.027624	0.026854	0.026822	0.071187	DoH
192.168.2(	176.103.13	50749	443	93828	767.137	101232	827.672	10458.12	102.2649	141.2455	0.799261	0.853132	0.724023	708.4659	26.61702	52.2879	48.83031	31.71966	0.389704	0.772748	0.509047	0.00117	0.0342	0.024387	0.021044	0.026981	0.293297	DoH
192.168.2(	176.103.13	50749	443	38784	320.6391	38236	316.1086	7300.294	85.44176	133.7153	1.570027	0.932978	0.638983	1358.911	36.86341	50.31611	39.77075	0.417528	0.858198	1.353607	0.732636	0.000785	0.028021	0.029238	0.026922	0.026855	0.248064	DoH
192.168.2(	176.103.13	50749	443	61993	561.0171	69757	631.2789	8499.283	92.19155	139.1235	0.817544	0.923333	0.66266	1118.135	33.43853	51.69373	34.8825	13.28093	1.508251	1.148758	0.646859	0.000411	0.020274	0.019925	0.019268	0.026918	0.097199	DoH
176.103.13	192.168.2(	443	50749	83641	1542.341	76804	1416.267	8052.746	89.73709	138.9134	0.83288	0.277627	0.645993	341.6966	18.48504	36.43562	49.82256	7.342519	-2.17261	1.573873	0.507334	0.079079	0.281209	0.02593	4.65E-05	2.10E-05	0.276133	DoH
192.168.2(	176.103.13	52491	443	54084	371.8117	63843	438.902	16074.98	126.7871	141.061	1.231852	0.686671	0.898811	1519.785	38.98442	52.96259	47.49244	53.80841	0.420949	-0.0217	0.736075	0.000643	0.025358	0.025075	0.026813	0.026903	-0.20561	DoH
192.168.2(	176.103.13	52742	443	1718	11054.63	7411	47686.76	234126	483.8657	351.1154	1.37403	0.614045	1.378082	0.001068	0.032679	0.068209	0.0609	0.087941	0.671006	-0.6038	0.479104	0.000155	0.012445	0.013211	0.010833	2.30E-05	0.573131	DoH
192.168.2(	176.103.13	52742	443	55	2036.962	66	2444.354	30.25	5.5	60.5	0	1	0.090909	0.000182	0.013501	0.013501	0.013501	0	0	1	1	0.00E+00	0	0.027001	0.027001	0.027001	-10	DoH
192.168.2(	176.103.13	52491	443	32285	723.0755	36193	810.6016	8371.95	91.49836	139.751	0.844311	0.937186	0.654724	193.9635	13.92708	23.25506	23.03529	0.644233	0.047339	1.623515	0.598884	0.000664	0.025772	0.021313	0.020787	0.000203	0.061237	DoH
192.168.2(	176.103.13	52742	443	55	1183.941	66	1420.73	30.25	5.5	60.5	0	1	0.090909	0.00054	0.023228	0.023228	0.023228	0	0	1	1	0.00E+00	0	0.046455	0.046455	0.046455	-10	DoH
192.168.2(	176.103.13	52742	443	163	3.631065	357	7.9527	168.25	12.97112	65	1.156415	0.848038	0.199556	377.2307	19.42243	33.65414	44.86306	44.86306	-1.73134	-0.57711	0.577119	9.00E-12	0.000003	0.027021	0.027021	0.027018	0	DoH
192.168.2(	176.103.13	52491	443	8090	88.57206	7805	85.45178	6158.022	78.47307	125.1575	1.382289	0.830316	0.626995	140.5086	11.85363	78.94149	81.96176	0	-0.76439	6.659689	0.150157	0.001596	0.039953	0.034821	0.026919	0.026899	0.593319	DoH
192.168.2(	176.103.13	52491	443	53598	441.7434	59634	491.4908	8309.003	91.15374	136.2599	1.555392	0.902431	0.668969	1017.683	31.90114	57.2504	55.49805	78.54598	0.164792	-0.66755	0.557221	0.000819	0.028622	0.026189	0.026823	0.026913	-0.06647	DoH
192.168.2(	176.103.13	52491	443	56840	505.8745	60236	536.0988	8218.835	90.65779	135.819	1.549311	0.902504	0.66749	1127.623	33.58009	34.36958	25.3958	72.24556	0.801705	-1.12793	0.97703	0.001352	0.036773	0.031305	0.026844	0.026952	0.36392	DoH
176.103.13	192.168.2(	443	52491	69610	514.6985	65101	481.3588	8353.123	91.39542	136.6237	1.56322	0.904025	0.668957	903.743	30.06232	50.51431	44.10633	20.5056	0.63947	0.998217	0.595125	0.266141	0.515889	0.079059	0.000042	2.10E-05	0.459501	DoH
192.168.2(	176.103.13	52491	443	35194	290.4132	37434	308.8972	7686.456	87.67244	136.5188	0.770554	0.941217	0.6422	1166.565	34.15502	56.05765	62.00261	76.86132	-0.52217	-0.6091	0.609284	0.000738	0.027169	0.027962	0.026912	0.026917	0.115979	DoH
192.168.2(	176.103.13	52491	443	78615	797.6961	81323	825.1738	7709.703	87.80491	135.4259	1.586218	0.92735	0.648361	707.6148	26.60103	54.34677	56.88736	29.66267	-0.28652	0.927938	0.489468	0.000778	0.027895	0.022306	0.020144	3.50E-05	0.232507	DoH
176.103.13	192.168.2(	443	52491	13565	410.3599	11354	343.4741	9409.972	97.00501	139.9944	1.577064	0.886494	0.692921	102.984	10.14811	12.20385	7.854821	33.02868	1.285668	-2.05209	0.831549	0.000429	0.020709	0.012648	0.000041	2.00E-05	1.82625	DoH
192.168.2(	176.103.13	54640	443	57076	496.7161	68354	594.8653	17033.22	130.5114	146.5304	0.74776	0.708983	0.890678	1178.686	34.332	46.67889	49.60508	0.054805	-0.2557	1.358036	0.735493	0.001067	0.032665	0.024944	0.025127	0.02696	-0.01685	DoH
192.168.2(	176.103.13	54640	443	42044	434.5628	44920	464.2889	7899.639	88.87992	133.7908	1.511841	0.897737	0.66432	488.3844	22.09942	39.33142	35.98585	1.445943	0.454162	1.71432	0.561877	0.0017	0.041234	0.033549	0.026931	0.026952	0.481463	DoH
192.168.2(	176.103.13	54640	443	61248	580.4434	63497	601.757	8029.013	89.60476	135.1517	1.545175	0.905663	0.662994	670.7841	25.8995	53.89672	56.66998	26.27162	-0.32123	1.066627	0.480539	0.001151	0.033924	0.028698	0.026865	0.026891	0.162068	DoH
192.168.2(	176.103.13	54640	443	27297	246.775	29436	266.1124	7805.691	88.34982	135.0786	1.564641	0.917699	0.654062	1472.847	38.37769	55.28208	41.52513	22.90654	1.075386	0.843603	0.694216	0.000801	0.028296	0.027009	0.026856	0.026912	0.016235	DoH
192.168.2(	176.103.13	54640	443	67491	554.2362	73742	605.5694	9569.622	97.82444	141.3744	0.839495	0.893175	0.691953	1079.693	32.85868	53.07471	52.74703	20.96734	0.029918	0.977135	0.619102	0.001652	0.040643	0.028948	0.026839	0.026905	0.155671	DoH
192.168.2(	176.103.13	54640	443	38991	320.4489	41627	342.113	8195.517	90.5291	135.4924	1.54069	0.90018	0.668149	1185.212	34.4269	67.15719	70.62596	89.83199	-0.30227	-0.65864	0.512632	0.001017	0.031897	0.029559	0.026934	0.027005	0.2469	DoH
192.168.2(	176.103.13	54640	443	72213	595.5716	80570	664.4953	10462.65	102.2871	139.9112	1.493185	0.839902	0.731086	1071.718	32.7371	50.99465	45.52109	15.50502	0.501593	1.08408	0.641971	0.001161	0.034066	0.02837	0.026885	0.026986	0.130788	DoH
192.168.2(	176.103.13	54640	443	30733	415.1167	32409	437.7547	7758.135	88.08027	132.6513	1.486755	0.89295	0.663999	650.7667	25.51013	33.52663	32.25093	1.671356	0.150023	1.24873	0.760892	0.001353	0.036781	0.033745	0.026932	0.026892	0.555735	DoH
192.168.2(	176.103.13	56611	443	35590	291.5291	44587	365.2264	21960.19	148.1897	146.3084	1.160169	0.622907	1.012859	552.312	23.50132	45.36756	41.00397	38.21969	0.557022	0.304148	0.51802	0.001381	0.037158	0.031246	0.026875	0.026911	0.35289	DoH
192.168.2(	176.103.13	56611	443	41539	431.0563	44577	462.5821	8937.776	94.53981	137.5655	1.541112	0.513704	0.687235	875.4395	29.58783	53.87872	60.85792	22.30918	-0.70764	1.066977	0.549156	0.000956	0.030926	0.026551	0.026848	0.026879	-0.02883	DoH
192.168.2(	176.103.13	56611	443	60659	499.8401	67897	559.4824	8887.787	94.27506	140.6521	0.848117	0.919141	0.670271	1147.148	33.86957	60.85415	75.55503	32.49268	-1.30213	0.837373	0.55657	0.001013	0.031829	0.027571	0.026862	0.026941	0.066819	DoH
192.168.2(	176.103.13	56611	443	30409	290.5247	30718	293.4768	8549.654	92.46434	133.7571	1.452142	0.862572	0.691285	547.7144	23.4033	70.56992	82.16181	54.19959	-1.48593	0.699488	0.331633	0.001226	0.035013	0.029797	0.026867	0.026908	0.251063	DoH
192.168.2(	176.103.13	56611	443	53440	568.2905	57703	613.624	8923.883	94.4663	138.7553	1.167251	0.897201	0.680812	1145.985	33.8524	47.36473	58.25138	0.849871	-0.96477	1.374049	0.714718	0.000407	0.020164	0.022742	0.025419	0.026951	-0.39827	DoH
192.168.2(	176.103.13	56611	443	39646	327.2182	40194	331.7411	10452.24	102.2362	137.4182	1.420776	0.815936	0.743978	1189.359	34.48709	57.70407	50.98765	8.825429	0.584255	1.417302	0.597654	0.001296	0.035995	0.03595	0.026918	0.026905	0.752785	DoH
192.168.2(	176.103.13	57679	443	1664	9702.284	7410	43205.49	239820.4	489.7146	362.96	1.402613	0.630898	1.349225	0.001068	0.032674	0.06773	0.064289	0.054882	0.315944	0.39322	0.482414	0.000551	0.023483	0.021766	0.026737	3.80E-05	-0.63505	DoH
192.168.2(	176.103.13	56611	443	27435	247.6404	29036	262.0917	7839.336	88.54002	133.1863	1.497164	0.894356	0.664783	989.83	31.46156	57.14953	61.23735	48.70869	-0.38979	0.26829	0.550513	0.001628	0.040349	0.0372	0.026916	0.026916	0.764613	DoH
192.168.2(	176.103.13	57679	443	55	2002.695	66	2403.233	30.25	5.5	60.5	0	1	0.090909	0.000189	0.013732	0.013732	0.013732	0	0	1	1	0.00E+00	0	0.027463	0.027463	0.027463	-10	DoH
192.168.2(	176.103.13	57679	443	55	2035.153	66	2442.183	30.25	5.5	60.5	0	1	0.090909	0.000183	0.013513	0.013513	0.013513	0	0	1	1	0.00E+00	0	0.027025	0.027025	0.027025	-10	DoH

Figure 5.10: CIRACICDOHBRW2020 sample data set

Figure 5.10 provides an extract of the CIRACICDOHBRW2020 data set. As in the above data sets, the majority of features such as the "rate of flow bytes sent" have numerical values.

The next section provides a summary of NIDSs cybersecurity data sets discussed in previous sections.

5.3.1.5 NIDSs cybersecurity data sets - summary

The NIDS data sets presented in this chapter contain bot related synthetic attacks, and therefore, they were all deemed relevant for the purpose of this thesis in discovering zero-day network intrusion attacks launched by bots. All of these data sets also have clear labels of types of attack, coupled with their features such as packet sizes with large number of instances. Although these data sets do not contain "zero-day" attacks, the author of this thesis created such an attack from known types of attack. In each NIDS data set, a zero-day attack type was created by manually combining malicious network traffic across the same features as indicated by Figures 5.3, 5.5, 5.7 and 5.9.

Each NIDS data set was implemented separately in the InternetBotDetector model for discovering zero-day network intrusion attacks. Having introduced the NIDS data sets that were in this thesis for discovering zero-day network intrusion attacks, the next section introduces the OSNs that were used to discover bot cyberattacks on OSNs.

5.3.2 OSN cybersecurity data sets

This thesis aimed specifically to classify benign and malicious OSN bots. A plethora of OSN bot data sets was reported in the literature, see bot repository [2] as an example, a well-known repository for 𝕏 bot data sets formed in 2015. The author of this thesis conducted a high-level assessment on data sets from this repository, published from 2015 and beyond which contained both benign and malicious bots. In this regard, only one data set by Oentaryo et al. [4] was found to satisfy this requirement at that time.

Having analysed the data set from Oentaryo et al. [4], it became clear to the author of this thesis that the sample size of about 453 samples was not large enough for the purposes of building an ML algorithm. Thus, the author of this thesis randomly selected three more OSN data sets that contain various bot cybercrimes published since 2019. In these, some of the data are not explicitly described to be containing benign or malicious bots. The summary of these data sets is found below in Table 5.5.

Author and year	**Data description**	**Comment**
Oentaryo et al. [4] - 2016.	Malicious bots - 80 and benign bots - 373.	Benign and malicious (spam) bots.
Yang et al. [5] - 2020.	Bots - 698.	Spam bots and fake follower bots.
Mazza et al. [6] - 2019.	Bots - 358.	Botnets - malicious bots.
Yang et al. [7] - 2019.	Bots - 17882.	Political bots - tweeting about politics. Malicious bots.

Table 5.5: Summary and number of instances of OSN bot data sets used in this thesis

[2] https://botometer.osome.iu.edu/bot-repository/index.html Last accessed: 29Apr2024.

The malicious bots in Table 5.5 launched various types of attacks in 𝕏. Oentaryo et al. [4] data set is based on spam attacks, Yang et al. [5] is based on spam and fake followers, Mazza et al. [6] is based on botnet attacks and Yang et al. [7] includes misinformation or fake news. Oentaryo et al. [4] extracted 𝕏 data sets using the 𝕏 API of active users from Singapore. In this regard, bots were labelled as benign or malicious based on their activities, such as tweeting patterns. Yang et al. [5] collected various data sets from the bot repository which served as a "ground truth" or labelled data set. In addition, these authors extracted more than 100,000 samples using 𝕏 API. Mazza et al. [6] extracted 𝕏 data sets using 𝕏 API of active users from Italy, and manually labelled malicious bots. Yang et al. [7] used data sets from the bot repository to manually label political bots.

These data sets cover the majority of known cybercrimes launched by bots in OSNs as demonstrated by the systematic literature conducted in this thesis. Next, extracts of these data sets are displayed to demonstrate that 𝕏 meta data is based on the same data structure of features, as demonstrated in the figures below.

id	screen_na	followers_count	friends_count	listed_count	favourites_count	verified	statuses_count	status.ret	status.favorite_count
5391152	oooyingqi	54	200	1	6	FALSE	4768	0	0
7648812	MandySG	762	0	25	0	FALSE	4200	0	0
8890532	jkupe	296	209	14	36	FALSE	1776	0	1
14120920	webmaste	7547	4828	117	514	FALSE	6090	0	0
14283304	nelsontan	851	987	18	0	FALSE	5779	0	0
14381494	damienoh	254	109	131	0	FALSE	10138	0	0
14489642	ShopBug	378	405	8	1	FALSE	6822	0	0
14740499	sginfomag	3154	20	80	2	FALSE	64155	2	3
15749326	techgoonc	1045	46	37	27	FALSE	3514	0	0
16011929	dreamrea	1243	411	31	314	FALSE	60396	2	4
16684029	cyborg500	196	77	1	5	FALSE	720	0	0
16859833	GuyBowel	3646	4062	50	7	FALSE	72273	0	1
17842245	gurrage	1870	2156	6	0	FALSE	3452	2	2
18220693	refinerym	8180	8837	129	0	FALSE	13181	1	9
18793777	daniel_ch	157	249	4	9	FALSE	1961	0	0
18795045	cellcity	239	121	9	8	FALSE	13360	0	0
18965199	reiepl	68	68	2	33	FALSE	1577	0	0
19345783	Devcrossi	4157	707	341	0	FALSE	7066	3	1
19761513	Mag_HR	23471	778	648	1363	FALSE	32205	0	0
19769058	wahizlife	1923	0	55	6	FALSE	123093	0	4
20572106	PlushAsia	896	186	21	14	FALSE	6750	1	1
21343476	ejection	325	1836	5	29	FALSE	1939	0	0

Figure 5.11: Oentaryo et al. [4] sample data set

Figure 5.11 provides an extract of the Oentaryo et al. [4] data set. This data set has a clearly labelled set of features that include the number of followers, friends, and statuses_count. In addition, most of these features are numeric.

user/id	user/id_str	user/name	user/screen_name	user/followers_count	user/friends_count	user/listed_count	user/favourites_count	user/verified	user/statuses_count
9.69E+17	9.69E+17	Random Goth Lyric Generator	goth_lyrics	23	8	1	2	FALSE	3792
2.402E+09	2.402E+09	Transiting Now	transitingnow	322	8	21	34	FALSE	407226
2.977E+09	2.977E+09	TheRiddlerBot	TheRiddlerBot	171	86	32	1933	FALSE	15668
4.426E+09	4.426E+09	ðŸŒŠðŸŒŠðŸŒŠðŸŒŠðŸŒŠðŸŒŠðŸŒŠðŸŒŠðŸŒŠðŸŒŠ	tiny_seas	3374	9	146	15	FALSE	5087
8.17E+17	8.17E+17	Wordfindatron	wordfindatron	7	2	4	1	FALSE	2813
7.42E+17	7.42E+17	NYPL Emoji Bot	NYPLEmoji	3968	15	110	0	FALSE	17685
9.98E+17	9.98E+17	CultureShip Generator	CultureShip_bot	277	25	6	5	FALSE	3046
1.02E+18	1.02E+18	ð"„ð•£ð•¥ ð•„ð•¤ð•–ð•¦ð•ž ð"ð•• ð•¥	art_museum_bot	55	1	10	0	FALSE	1961
7.19E+17	7.19E+17	Wizard Generator	WizardGenerator	177	0	36	52	FALSE	9166
3.305E+09	3.305E+09	Villanelle Bot	VillanelleBot	106	1	20	229	FALSE	10279
3.749E+09	3.749E+09	OMG Movies!	omg_movies	15	1	14	0	FALSE	1246
1.05E+18	1.05E+18	OTPbot	ishipitbot	13	7	2	7	FALSE	1951
4.332E+09	4.332E+09	Lotostodorono	LatostadoraBot	101	81	4	20	FALSE	23227
1.10E+18	1.10E+18	Wikipedia Movie Plot Bot	WikiPlotBot	22	1	1	0	FALSE	445
9.20E+17	9.20E+17	Whiny Genealogist	whinygenealogy	19	0	4	0	FALSE	2513
8.38E+17	8.38E+17	CozinhaBot	CozinhaBot	3274	9	29	35	FALSE	19812
1.04E+18	1.04E+18	The Trumpster Fire	USDumpsterFire	8	24	1	2	FALSE	2505
1.04E+18	1.04E+18	Philly Tax Abatements	phl_abatements	436	1	5	5	FALSE	3267
4.887E+09	4.887E+09	Jeorkov	jeorkov	36	3	2	2	FALSE	8462
9.24E+17	9.24E+17	Good Dogs of NYC	GoodDogsOfNYC	56	2	4	1	FALSE	9421
9.04E+17	9.04E+17	bot de colores	bot_de_colores	9109	7	46	478	FALSE	39977
9.64E+17	9.64E+17	Hypno Bot	hypno__bot	31	2	12	0	FALSE	4003
1.01E+18	1.01E+18	Every Herd Bot	everyherd_bot	14	1	0	0	FALSE	6400
9.27E+17	9.27E+17	AntiqueAppraisalBot	AppraisalBot	14	8	4	4	FALSE	345

Figure 5.12: Yang et al. [5] sample data set

Figure 5.12 presents an extract of the Yang et al. [5] data set. Once again, this data set contains a complete set of features that are numeric in nature. The same findings apply to the data sets in Figures 5.13 and 5.14.

user/id	user/verified	user/entit	user/entit	user/follo	user/retw	user/liste	user/statu	user/desc	user/friends	user/screen_name	user/favo	user/name
86166567	FALSE	43	66	42	0	0	101	Candy girl	240	greenhomunculus	515	Kathy Carrington
108704199	FALSE	18	41	16	0	0	38	I want you	0	seizastar	187	Juliet Bush
2165940152	FALSE	50	73	7	0	0	32	Cute lady.	0	chechelala9876	178	Louise Creighton
196846880	FALSE	40	63	99	0	0	145	Cosplay m	500	MischaBruce	838	Gladys Ellington
884237364	FALSE	34	57	59	0	0	39	Blogger /	419	cucum87	628	Sue Marlow
2424529777	FALSE	15	38	15	0	0	37	Rate me o	183	Bbennana	156	Betty Taft
1650916326	FALSE	34	57	56	0	0	119	Cosplayer	0	alfredoaleman7	572	Peggy Eddington
120360548	FALSE	17	40	5	0	0	14	You want	1	luv_unguviolet	70	Sara Enderson
8.71E+17	FALSE	50	73	48	1	0	114	Cute lady.	337	vdokii	548	Bernice Fisher
313783257	FALSE	35	58	15	0	0	21	Twerk dar	75	gilja90	212	Wilma Dutton
132787155	FALSE	31	54	176	0	2	289	Cosplayer	0	FlyingJenny7	1074	Pamela Enderson
7.42E+17	FALSE	42	65	14	0	0	22	Cosplay m	136	hedingjian1	140	Suzanne Ward
1077404096	FALSE	42	65	21	0	0	78	Candy girl	21	anwar_lala	307	Mary Gimson
2154795135	FALSE	31	54	13	0	0	42	Do you lik	0	YTBRsBGS	76	Bonnie Calhoun
1394465665	FALSE	12	35	11	0	0	16	Come in ð	94	Mohammed903	96	Lucy Lamberts
2657551658	FALSE	36	59	5	0	0	38	Do you lik	0	doxik159	236	Barbara Gill
83124019	FALSE	38	61	277	0	7	726	Cosplay m	0	adyputraID	810	Wendy Berrington
4257661033	FALSE	12	35	15	0	0	66	Come in ð	0	wwb035	381	Beryl Haig
1868979648	FALSE	18	41	9	0	0	8	Come to n	0	2Leuts	84	Carmen Donovan
901274646	FALSE	18	41	35	0	0	58	I want you	0	MontyKing7	312	Alyssa Baldwin
1089449269	FALSE	12	35	14	1	0	48	Come in ð	0	DittrichLisa	179	Claire Watson
1523047624	FALSE	26	58	12	0	0	34	Do you lik	4	chtlcv	208	Ruby Paterson

Figure 5.13: Mazza et al. [6] sample data set

user/id	user/id_st	user/nam	user/followers_count	user/friends_count	user/liste	user/favo	user/verified	user/statuses_count
3.02E+09	3.02E+09	patatavis	218	285	0	2305	FALSE	3477
7.54E+17	7.54E+17	#1DMITAN	256	358	0	7561	FALSE	5614
9.02E+17	9.02E+17	Anna 056C	287	346	3	6443	FALSE	5545
2.98E+09	2.98E+09	Hanna	266	354	0	1202	FALSE	6022
8.25E+17	8.25E+17	Zaffiro Blu	286	564	0	2511	FALSE	3915
9.71E+17	9.71E+17	Marghe	26	48	0	15244	FALSE	2175
1.01E+18	1.01E+18	1D	15	24	0	52	FALSE	496
1E+18	1E+18	50 sfumat	19	52	0	37	FALSE	1469
3.41E+09	3.41E+09	Peugeot C	604	134	32	8727	FALSE	8554
6.17E+08	6.17E+08	gerry fuck	286	527	11	46531	FALSE	35242
8.17E+17	8.17E+17	Chiara	229	159	2	19618	FALSE	9495
9.79E+17	9.79E+17	Elena	908	685	1	22759	FALSE	12680
3.07E+09	3.07E+09	Cappellaic	314	642	1	13648	FALSE	16895
1.45E+09	1.45E+09	Laura Brog	856	2713	213	1828	FALSE	53970
8.5E+17	8.5E+17	romevatic	543	2247	1	4001	FALSE	4383
7.58E+17	7.58E+17	NightSun	349	242	4	15315	FALSE	15138
2.92E+09	2.92E+09	filippotrav	148	76	0	2190	FALSE	4801
1E+18	1E+18	ðŸŒ™sÏfg	6037	5330	1	4656	FALSE	5296
8.39E+17	8.39E+17	antonello	7	10	0	4111	FALSE	4111
2.62E+09	2.62E+09	v;,	642	430	2	7072	FALSE	8350
4.61E+09	4.61E+09	One Direc	216	214	0	2266	FALSE	2455
7.27E+17	7.27E+17	Sonia	93	182	25	6826	FALSE	6181

Figure 5.14: Yang et al. [7] sample data set

The next section provides a summary of OSN cybersecurity data sets discussed in the previous section.

5.3.2.1 OSN cybersecurity data sets - summary

As was shown in Table 5.5, the data set from Oentaryo et al. [4] is the main data set that contains clearly labelled benign and malicious bots. Given that this thesis adopts semi-supervised ML algorithms, Oentaryo et al. [4] data set was treated as a "labelled" data set. However, this data set was not large enough for the training of an ML algorithm. Several options can be applied to address this problem, such as creating synthetic data without labels or using original unlabelled data, i.e., [5], [6] and [7] data sets. In the current chapter, the latter option was chosen as it is more reflective of a real-world scenario where the original data extracted from an OSN platform is unlabelled. The question was whether these data sets could be combined into a single "unlabelled" data set. The answer was yes, given that the 𝕏 data sets are based on the same meta data structure, see Figure 5.11, 5.12, 5.13 and 5.14. To confirm this, the author of this thesis performed the MannWhitney U test to demonstrate that no bias was introduced by combining the data sets, see the results in Table B.1 in the Appendix B.

In Chapter 3, state-of-the-art methods and significant features identified by respective authors for discovering malicious bots in OSNs were indicated. Based on these results, the author of this thesis identified the most used features for discovering malicious 𝕏 bot accounts. Table 5.6 summarises these features.

No.	Feature name	Description	Min	Max
1	Account age (in days)	No. of days of accounts existence [202]	2555	5110
2	Screen_name length	No. of characters in the screen name [203]	6	16
3	Favourites_count	No. of tweets liked by a user [205]	0	28517
4	URL_count	No. of URLs in the user profile [213]	0	1
5	Hashtag_count	No. of tweets with # key phrase [4]	0	15
6	Lists_count	No. of pinned favourite lists [205]	0	736
7	Statuses_count	No. of tweets an account has [4]	1	1125425
8	Status.retweet_count	No. of retweets an account has [4]	0	244253
9	Status.reply_count	No. of replies per user's status [4]	0	711
10	Friends_count	No. of users an account is following [128]	0	58508
11	Followers_count	No. of followers an account currently has [128]	0	55482
12	Status.favourite_count	No. of likes per user's status [4]	0	22

Table 5.6: Commonly used 𝕏 features used to discover malicious bots

Once more, the OSN data sets presented in Table 5.5 are real-world bot cyberattacks that include spamming ([4] data set) and social engineering ([5] data set). Thus, they are deemed usable and relevant to the problem at hand.

5.4 Chapter summary

From a cybersecurity perspective, a ML-based cybersecurity solutions should ideally be trained on real-world attack data, so that such solutions can produce reliable results. However, due to the sensitive and private nature of real-world attack data, organisations can be challenged to obtain such data that is sufficiently large to train a ML algorithm. The CySecML methodology provides a framework to overcome this challenge through using synthetic data. The OSN cybersecurity data sets used in this thesis are real-world attack data, whereas, the NIDSs cy-

bersecurity data sets are based on synthetically generated attacks. The data extraction block of the CySecML methodology aims to cater for both scenarios of collecting cybersecurity data sets for the purposes of training ML-based cybersecurity solutions. The author of this thesis opted to use publicly available data sets, mainly for benchmarking reasons, even though some of these data sets can be considered outdated. The next section examines data quality of the cybersecurity data sets discussed in this chapter.

Chapter 6

Data quality checks on cybersecurity data sets

6.1 Introduction

This chapter examines the cybersecurity data sets introduced in the previous chapter in terms of their quality. This chapter aims to identify the important data quality characteristics to be considered on cybersecurity data sets. Ensuring data quality is a key step for the data preparation component of the CySecML methodology. The cybersecurity data sets used were examined to ensure that they are suitable for the problem at hand, namely to discover bot cyberattacks such as social engineering attacks in OSNs, and zero-day network intrusion attacks in NIDSs. Figure 6.1 illustrates the focus of this chapter on the CySecML methodology.

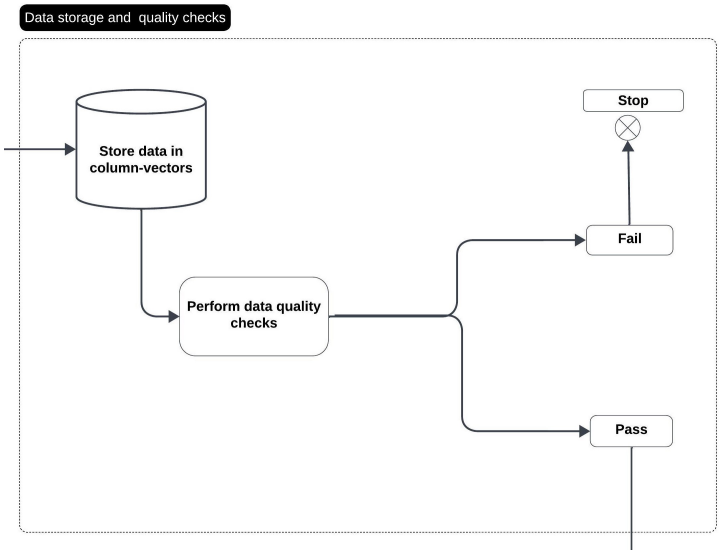


Figure 6.1: Data quality checks - taken out from Figure 4.2

6.2 Related work and background

6.2.1 Defining data quality for cybersecurity machine learning models

The process of ensuring that a data set is suitable for a particular ML problem is referred to as data preparation [270]. Data preparation involves inter alia data extraction, data cleaning and the assurance of data quality characteristics [270, 271]. The data extraction step refers to the methods for collecting data to be used in the experiments of an ML algorithm [270]. Once data has been extracted, it must be examined to determine whether this data set is suitable for training an ML algorithm to solve a particular problem [237]. The data quality characteristics part aims to address this question.

Data quality is widely described in the literature across different fields of study and different industries. For instance, in the financial sector, the International Monetary Fund [1] describes data quality as a measure of the condition of data characteristics such as accessibility, integrity, serviceability, methodological soundness, accuracy, and reliability. Mahanti [272] suggests that judging the quality of data requires an examination of its characteristics against its use. The characteristics to be considered are mainly informed by the requirements of the organisation or application system that uses this data. For example, consider an application system such

[1] https://dsbb.imf.org/dqrs/DQAF Last accessed: 25Apr2024.

as the InternetBotDetector model that aims to discover malicious bot activities on IAPs using ML algorithms. The ML algorithms used in this case require accurate data from IAPs, as this data depicts the activities of malicious bots for training purposes. Accuracy is therefore a data characteristic of importance in this application system.

Mahanti [272] expands the IMF list and argues that the most common characteristics of data quality include accuracy, reliability, legitimacy, consistency, relevance, flexibility, currency, completeness, availability, uniqueness, and timelessness. These characteristics of data quality are not defined explicitly for an ML cybersecurity problem, even though some of them may apply to it. For example, the currency characteristic, which refers to the monetary value attached to data, is not applicable to an ML cybersecurity problem. Cai et al. [11] presented a framework of data quality characteristics that consists of availability, usability, reliability, relevance, and presentation quality. From a cybersecurity perspective, it is crucial that a ML-based cybersecurity solution is not trained on corrupt data as this can result in unintended outcomes.

Ridzuan et al. [12] presented a taxonomy of data quality characteristics in big data platforms that is based on accessibility - that includes availability of data, contextual - that includes relevance and validity of data, intrinsic - that includes reliability and usability of data and representational - that includes interpretability and readability of data. Cai et al. [11] data quality characteristics are consistent with the taxonomy of data quality characteristics presented by Ridzuan et al. [12] for big data platforms. The cybersecurity data sets presented in this thesis are examined in accordance with Cai et al.'s [11] and Ridzuan et al. [12] framework of data quality, which is discussed below.

Data quality characteristic	Definition	Quantitative?
Availability	Accessibility of data conditional to security privilege	No
Usability	Data is aligned to its intended purpose	No
Reliability	Correctness and completeness of data	No
Relevance	Data is applicable for intended purpose and meaningful	No
Data size	Amount of data records	Yes
Presentation quality	Ease of readability and interpretability of data	No

Table 6.1: Data quality checks for ML-based cybersecurity problems adopted from [11, 12]

- *Availability* refers to the extent to which data is easily made available and accessible to systems and individuals who have the right level of access to obtain such data. Therefore, availability of cybersecurity data is an important aspect for consideration. In the context of the InternetBotDetector model component, the hypothesis is that data from an IAP is stored in a private storage repository such that the InternetBotDetector model can easily consume this data. This data quality characteristic is not quantitative.

- *Usability* refers to the extent to which data meets the requirements for solving the problem at hand. For example, consider the problem of discovering malicious bots in 𝕏 using ML algorithms. The question to be answered is whether the cybersecurity data set used depicts malicious bot behaviour. The cybersecurity data sets from 𝕏 and NIDS used in this thesis depict real-world bot cybercrimes and thus can be used for illustrative purposes to deal with the problem at hand. This data quality characteristic is not quantitative.

- *Reliability* refers to whether one can trust the data in terms of accuracy and completeness. Therefore, reliability of cybersecurity data sets is key in the context of cybersecurity ML-based solutions. In addition, the cybersecurity data sets from 𝕏 and NIDS used in this thesis are accurate and complete as they are clearly labelled, with a complete set of features that characterise bot behaviours. This data quality characteristic is not quantitative.

- *Relevance* of data is closely linked with the usability of data, with the exception that relevance asks the question as to why this data is needed in the first place? It is crucial to gather data that is relevant to the problem at hand which is to discover bot cybercrimes using ML algorithms. Data that is accurate but not relevant, is as good as redundant. For example, the 𝕏 data sets used in this thesis characterise bot behaviour in terms of followers, friends, tweets, features, and so on. This information will be key when classifying benign and malicious 𝕏 bots. This data quality characteristic is not quantitative.

- *Data size* refers to the number of data records in a data set. In the context of building ML-based cybersecurity solutions, data records must be sufficiently large, e.g., data records must be ten times the number of features or attributes of that data set [236, 73]. This data quality characteristic is quantitative.

- *Presentation quality* refers to the manner in which cybersecurity data is presented in terms of readability and interpretability. The cybersecurity data sets used in this thesis had clear descriptions of benign and malicious bot behaviours with relevant features on each data

set, thus, easy to read and understand the data. See Figure 5.4 and 5.11 for example. This data quality characteristic is not quantitative.

Having defined the data quality checks for ML-based cybersecurity problems, the next step is to examine the cybersecurity data sets used in this thesis.

6.3 Experimental work

The NIDS cybersecurity data sets used in this thesis are examined in terms of the data quality characteristics presented in Table 6.1. The IoT Intrusion-2020 data set is examined next.

6.3.1 IoT Intrusion-2020 - data quality checks

- Availability: the IoT Intrusion-2020 data set complies as follows to this criterion - this is a publicly available data set, and easily accessible on the Internet.
- Usability: the IoT Intrusion-2020 data set complies as follows to this criterion - this data set contains benign network traffic and a variety of malicious network traffic with a large number of instances, this data set can be useful for the purposes of training and testing ML-based cybersecurity solutions for discovering cyberattacks such as zero-day network intrusion attacks.
- Reliability: the IoT Intrusion-2020 data set complies as follows to this criterion - given that the cyberattacks such as flooding attacks considered in this data set are real-world cyberattacks. In addition, this data set is complete and accurate as it contains fully labelled features and values that characterise benign and malicious network traffic.
- Relevance: the IoT Intrusion-2020 data set complies as follows to this criterion - this data set contains more than 80 features that include the number of packets and flow duration of benign and malicious network traffic. This type of information is needed to train ML-based cybersecurity solutions to correctly learn to differentiate between benign and malicious network traffic.
- Data size: the IoT intrusion-2020 data set complies as follows to this criterion - this data set has samples that are ten times the number of its features.

- Presentation quality: the IoT Intrusion-2020 data set complies as follows to this criterion - this data set has clearly labelled features and network traffic types. In addition, the meaning of features in this data set are described.

The UNSW-NB15 cybersecurity data set is examined next.

6.3.2 UNSW-NB15 - data quality checks

- Availability: the UNSW-NB15 data set complies as follows to this criterion - this is a publicly available data set, and easily accessible on the Internet.
- Usability: the UNSW-NB15 data set complies as follows to this criterion - this data set contains benign network traffic and nine malicious network traffic with a large number of instances. These intrusion attacks are based on real-world cyberattacks such as DoS. Therefore, this data set can be useful for the purposes of training and testing ML-based cybersecurity solutions for discovering cyberattacks such as zero-day network intrusion attacks.
- Reliability: the UNSW-NB15 data set complies as follows to this criterion - given that this data set is complete and accurate as it contains fully labelled features and values that characterise benign and malicious network traffic.
- Relevance: the UNSW-NB15 data set complies as follows to this criterion - this data set contains more than 44 features that include the number of duration and transaction protocol of benign and malicious network traffic. This type of information is useful to train ML-based cybersecurity solutions to correctly learn to differentiate between benign and malicious network traffic.
- Data size: the UNSW-NB15 data set complies as follows to this criterion - this data set has samples that are ten times the number of its features.
- Presentation quality: the UNSW-NB15 data set complies as follows to this criterion - this data set has clearly labelled features and network traffic types. In addition, the meaning of features in this data set is described.

The CICDDOS2019 cybersecurity data set is examined next.

6.3.3 CICDDOS2019 - data quality checks

- Availability: the CICDDOS2019 data set complies as follows to this criterion - this is a publicly available data set, and easily accessible on the Internet.
- Usability: the CICDDOS2019 data set complies as follows to this criterion - this data set contains benign network traffic and various types of botnet DDoS attacks. Therefore, this data set can be useful for the purposes of training and testing ML-based cybersecurity solutions for discovering cyberattacks such as zero-day network intrusion attacks.
- Reliability: the CICDDOS2019 data set complies as follows to this criterion - given that this data set is complete and accurate as it contains fully labelled features and values that characterise benign and malicious network traffic.
- Relevance: the CICDDOS2019 data set complies as follows to this criterion - this data set contains more than 80 features that include the number of protocol and flow duration of benign and malicious network traffic. This type of information is useful to train ML-based cybersecurity solutions to correctly learn to differentiate between benign and malicious network traffic.
- Data size: the CICDDOS2019 data set complies as follows to this criterion - this data set has samples that are ten times the number of its features.
- Presentation quality: the CICDDOS2019 data set complies as follows to this criterion - this data set has clearly labelled features and network traffic types. In addition, the meaning of features in this data set are described.

The CIRACICDOHBRW2020 cybersecurity data set is examined next.

6.3.4 CIRACICDOHBRW2020 - data quality checks

- Availability: the CIRACICDOHBRW2020 data set complies as follows to this criterion - this is a publicly available data set, and easily accessible on the Internet.
- Usability: the CIRACICDOHBRW2020 data set complies as follows to this criterion - this data set contains benign network traffic and malicious DoH network traffic. Therefore, this data set can be useful for the purposes of training and testing ML-based cybersecurity solutions for discovering cyberattacks such as zero-day network intrusion attacks.

- Reliability: the CIRACICDOHBRW2020 data set complies as follows to this criterion - given that this data set is complete and accurate as it contains fully labelled features and values that characterise benign and malicious network traffic.
- Relevance: the CIRACICDOHBRW2020 data set complies as follows to this criterion - this data set contains about 35 features that include source and destination IP address of benign and malicious network traffic. This type of information is useful to train ML-based cybersecurity solutions to correctly learn to differentiate between benign and malicious network traffic.
- Data size: the CIRACICDOHBRW2020 data set complies as follows to this criterion - this data set has samples that are ten times the number of its features.
- Presentation quality: the CIRACICDOHBRW2020 data set complies as follows to this criterion - this data set has clearly labelled features and network traffic types. In addition, the meaning of features in this data set are described.

The next section examines the OSN cybersecurity data sets in terms of the data quality characteristics presented in Table 6.1.

6.3.5 OSN data sets - data quality checks

- Availability: the OSN data sets in Table 5.5 comply as follows to this criterion - this is a publicly available data set, and easily accessible on the Internet.
- Usability: the OSN data sets in Table 5.5 comply as follows to this criterion - these data sets contain benign and various types of malicious bots. Therefore, this data set can be useful for the purposes of training and testing ML-based cybersecurity solutions to differentiate between benign and malicious bots.
- Reliability: the OSN data sets in Table 5.5 comply as follows to this criterion - given that these data sets are complete and accurate as they contain fully labelled features and values that characterise benign and malicious bots.
- Relevance: the OSN data sets in Table 5.5 comply as follows to this criterion - these data sets contain features such as friends_count and followers_count that can be useful when building ML-based cybersecurity solutions.

- Data size: the OSN data sets in Table 5.5 comply as follows to this criterion - this data set has samples that are ten times the number of its features.
- Presentation quality: the OSN data sets in Table 5.5 comply as follows to this criterion - these data sets have clearly labelled features and their meaning described.

6.4 Chapter summary

In recent years, organisations have access to big data mainly driven by technologies such as IoT, hence, the quality of data has become a crucial factor for consideration when designing ML-based cybersecurity solutions. Existing ML-based cybersecurity methodologies overlook the guidelines for checking data quality on cybersecurity data sets. From a cybersecurity perspective, this thesis found availability, usability, reliability, relevance, data size, and presentation quality to be important data quality characteristics to be considered when designing ML-based cybersecurity solutions. In as much as these data quality characteristics will assist organisations to effectively identify data sets that are of quality. However, a shortcoming is that the majority of these data quality characteristics are not quantifiable. The next chapter deals with data cleaning and feature selection block, together with the machine learning and decision-making block.

Chapter 7

Feature selection and machine learning

7.1 Introduction

The previous chapter demonstrated that the NIDS and OSN cybersecurity data sets used in this thesis are of quality. The next blocks in the CySecML methodology are to implement feature selection and machine learning algorithms. In this thesis, Benford's law (BL) is proposed as a feature selection method to identify significant features that can assist in the intelligent discovery of cyberattacks launched by bots. Figure 7.1 illustrates the focus of this chapter on the CySecML methodology.

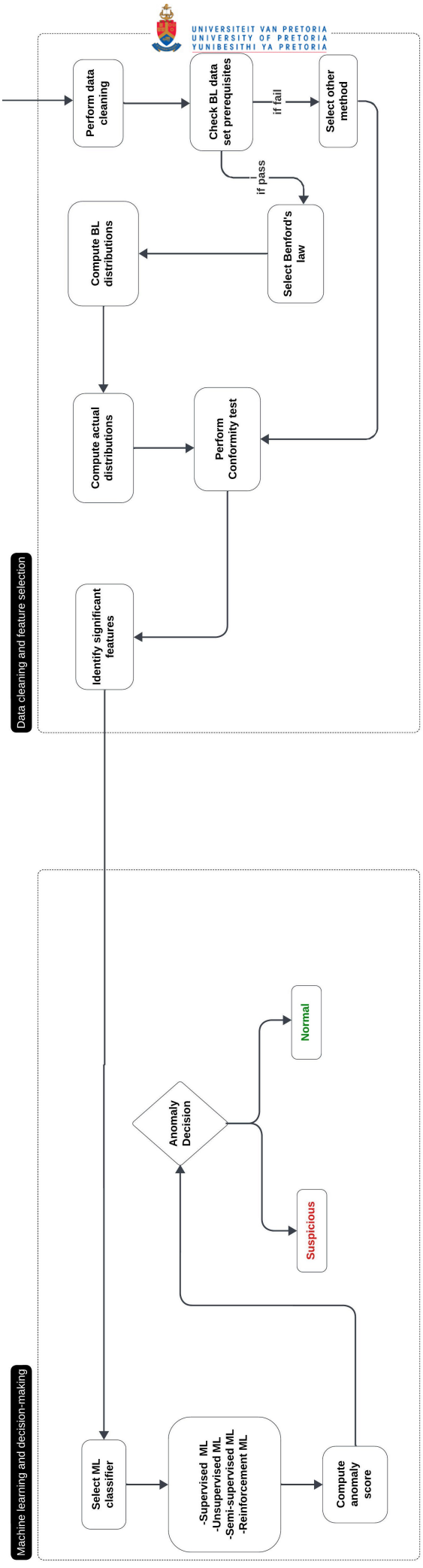


Figure 7.1: Feature selection and machine learning blocks - taken out from Figure 4.2

7.2 Related work and background

Feature selection and data cleaning block is discussed first, followed by machine learning and decision-making block.

7.2.1 Feature selection

Various attributes and features describe user activities on big data platforms, such as OSNs and NIDSs [2, 273, 274]. The information contained in attributes and features can be used for analysis, including input variables into ML algorithms [1, 246, 247]. However, not all features are useful or significant in the design of ML algorithms for the purposes of discovering a particular activity [238, 244]. Given the large number of features and attributes found in NIDS and OSN platforms, the question is - which features should be monitored to intelligently discover malicious bot activities, which features are less significant, and which features are redundant? Therefore, it is of utmost importance that significant features are identified prior to designing ML-based discovery models, because this usually affects the accuracy, overall performance, interpretability of model results, and computational cost [244, 275, 276].

A feature or attribute is considered significant if it can differentiate between two or more data classes [1]. For example, consider a data set from $\mathbb{X}$ that consists of various features describing human and malicious bot activities. A feature such as the number of tweets *liked* by other users was shown in [189] to distinctively differentiate between human and malicious bot data sets. In this regard, tweets by malicious bots were less liked than those made by humans. Thus, this feature was considered significant [189].

A plethora of feature selection methods exists, particularly for NIDS and OSN data types [207]. Feature selection methods can be broadly categorised under three main headings: filter, wrapper, and embedded [277, 244, 247]. Given a large set of input features $X = \{x_1, x_2, \cdots, x_n\}$, the aim is to identify a sub-set of $X' \subseteq X$ of significant features.

7.2.1.1 Filter methods

Filter methods, also known as feature ranking methods, identify significant features based on the correlation between a feature and a dependent variable by using statistical methods [207]. The filter approach assumes that features ranked higher have a greater impact on an ML algorithm's performance [277, 278]. However, the process of selecting significant features is independent of the ML algorithm used [247]. Examples of filter methods include the F-statistic [277], information gain (IG) [276], and correlation coefficient feature selection [244]. Filter methods are generally more computationally efficient than wrapper methods [247, 244].

7.2.1.2 Wrapper methods

Wrapper methods, also known as performance-based feature ranking methods, identify significant features based on the performance of an ML algorithm [278, 148]. Typically, there are two approaches for implementing wrapper methods: (i) evaluating the performance of a ML algorithm using all possible input features X, which becomes the base case. Thereafter, each feature is iteratively removed from X, and the performance of an ML algorithm is evaluated on both training and testing data sets. By using certain performance criteria, a decision is then made to maintain or remove the feature. Secondly, adding each feature at a time into an ML algorithm and evaluating the performance; this becomes the base case. Each feature is again iteratively added at a time, and the performance of the ML algorithm is evaluated on both the training and testing data sets [277, 244]. Examples of wrapper methods include sequential feature selection and recursive feature elimination (RFE). Wrapper methods are generally computationally expensive because they perform at least 2^n experiments, particularly when n is large. A wrapper approach would, for example, be infeasible for the IoT intrusion data set [8] used in this thesis, as it has $n = 86$ features.

7.2.1.3 Embedded methods

Embedded methods identify significant features by optimising an objective function, ML algorithm performance, or both [277, 244]. Thus, embedded methods combine the filter and wrapper techniques [244]. This approach was disregarded in this thesis because the cybersecurity data sets used have a relatively large number of features, thus potentially increasing computational

cost. Pilnenskiy et al. [206] provided Python code - which is freely accessible - for various filter, wrapper and embedded feature selection methods.

Examples of most used feature selection and extraction methods for OSN and NIDS data sets include information gain (IG) [274], principle component analysis (PCA) [207], correlation coefficient [246], and SMOTE [206]. The aim of this thesis was to effectively identify features that can assist in the intelligent discovery of anomalies, e.g., malicious bots in high-dimensional big data platforms, by using a probability distribution approach. Thus, Benford's law (BL) is proposed as a feature selection method, given its ability to identify anomalies on big data platforms [279, 280]. In the next section, BL is introduced as a feature selection method to assist ML algorithms used in discovering cyberattacks launched by bots on OSNs and NIDSs.

7.2.2 Benford's law

This section provides the background history of Benford's law (BL), its definition, and application. It then proceeds to motivate why BL was considered as a feature selection method for discovering bot cyber threats in OSNs and NIDSs.

7.2.2.1 History of Benford's law

Background

The history of BL began in 1881 when astronomer Simon Newcomb observed that in logarithmic tables, pages of logarithms beginning with smaller digits were worn out compared to those containing logarithms beginning with higher digits [281, 282]. Newcomb published an article called *Note on the Frequency of Use of the different digits in Natural Numbers* [283], where he indicated that numbers chosen from the logarithmic reference are not evenly distributed. Newcomb's paper was not well received by academics because he could not prove his theory. Only about 50 years later, in 1938, when physicist Frank Benford published a paper entitled *The law of Anomalous Numbers* [284], did Newcomb's theory receive attention.

Benford's law in practice

Benford [284] indicated that Newcomb's theory of leading digits was observable in several data sets, including population size, river size, and mathematical constants. Although Benford illustrated the expected distribution of the leading digits of a natural set of numbers, he could not mathematically prove this theory. In 1961, Pinkham [285] proved that BL is scale invariant. This means that the distribution of the leading digits is not affected by the unit of measurement, e.g., measuring length in terms of inches or metres. Vladimir Arnold, who generalised Benford's law and provided a mathematical proof in 1999, demonstrated that Benford's law distributions were expected to obey a geometric distribution [286, 287]. Although this theory has been known for decades, it has had little impact in practice.

Nigrini [288] suggested that BL can be used to detect financial fraud. In 1997, Nigrini [288] investigated the tax declaration of several states in the United States of America and discovered that the treasury of the office of Arizona was involved in fraud activities. Over the years, BL has been proposed to discover abnormal or suspicious activities in various domains that include OSNs [219, 220], computer networks [182, 183], credit card fraud [280], money laundering [289], and accounting fraud [282] among others.

7.2.2.2 Benford's law - description

The intuition behind BL can be explained using scientific notation, which states that for any positive real number $x \in \mathbb{R}^+$ it can be written in scientific notation as $x = S(x) \times 10^k$ where $S(x) \in [0, 1)$ is called a **significand** and k is an integer called the exponent [13]. For example, consider a feature on OSNs such as "friend_count" that has a value $x = 801$ for a particular user. This number of 801 can be written as 8.01×10^2 in scientific notation where 8.01 is the significand, 2 is the exponent, and **8** is the first significant leading digit (FSLD). The main advantage of analysing leading digits as opposed to full digits is that every number has a unique FSLD, i.e., $1, 2, \cdots, 9$; thus, one can compare the distributions of the leading digits for multiple data sets despite the differences in the magnitude of numbers, which is a like-for-like comparison. Different mathematical methods such as the central limit theorem prove BL - see [13, 290].

In the next section, BL propositions are stated.

7.2.2.3 Benford's law propositions

Proposition 1 *Benford's law for the leading digits: a data set is said to satisfy Benford's law for the leading digit if the probability of observing a digit d is* $\log_{10}\left(1+\frac{1}{d}\right)$.

Proposition 2 *Strong Benford's law for the leading digits: a data set is to satisfy Strong Benford's law for the leading digits if the probability of observing a significand s in $[1,s)$ is* $\log_{10} s$.

The mathematical proofs of the above propositions are found in [13]. The BL probability distribution was computed using the following properties.

7.2.2.4 Benford's law properties

Property 1: The First significant leading digit (FLSD)

Let D be a positive real number on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$. The logarithmic density function for the first leading digit is given by;

$$\mathbb{P}(D=d) = \log_b(d+1) - \log_b(d) = \log_b\left(1+\frac{1}{d}\right) \tag{7.1}$$

where $b=10$ (other basis also applicable) and $d \in \{1,2,\ldots,9\}$. Examples

$$\mathbb{P}(D=1) = \log_{10}\left(1+\frac{1}{1}\right) = 0.30103$$
$$\mathbb{P}(D=9) = \log_{10}\left(1+\frac{1}{9}\right) = 0.04576$$

The above property can be extended, for example, the probability that a number starts with leading digits of 2,1,6 is

$$\mathbb{P}(D=216) = \log_{10}\left(1+\frac{1}{216}\right) = 0.0020 = 0.2\%$$

Property 2: The n^{th} significant leading digit

The logarithmic density function of the leading digit $d \in \{0,1,2,\ldots,9\}$ is observed in the $n^{th}(n>1)$ position is given by

$$\mathbb{P}(d \text{ in } n^{th} \text{ position}) = \sum_{k=10^{n-2}}^{10^{n-1}-1} \log_{10}\left(1+\frac{1}{10k+d}\right) \tag{7.2}$$

Example 1: The probability that 0 is observed on the second position ($n = 2$) is given by

$$\sum_{k=1}^{9} \log_{10}\left(1 + \frac{1}{10k+0}\right) = \log_{10}\left(1 + \frac{1}{10}\right) + \cdots + \log_{10}\left(1 + \frac{1}{90}\right) = 0.11968$$

Example 2: The probability that 9 is observed on the fourth position ($n = 4$) is given by

$$\sum_{k=100}^{999} \log_{10}\left(1 + \frac{1}{10k+9}\right) = 0.09982$$

In the table below, the author of this thesis summarises Benford's law expected n^{th} digit position.

Digit	1st	2nd	3rd	4th
0	N/A	0.11968	0.10178	0.10018
1	0.30103	0.11389	0.10138	0.10014
2	0.17609	0.10882	0.10097	0.10010
3	0.12494	0.10433	0.10057	0.10006
4	0.09691	0.10031	0.10018	0.10002
5	0.07918	0.09668	0.09979	0.09998
6	0.06695	0.09337	0.09940	0.09994
7	0.05799	0.09035	0.09902	0.09990
8	0.05115	0.08757	0.09864	0.09986
9	0.04576	0.08500	0.09827	0.09982

Table 7.1: The expected digit frequencies of BL from first to fourth digits adopted from [13]

It can be observed from Table 7.1 and Figure 7.2 that the fourth digit's distribution is uniform at about 10%.

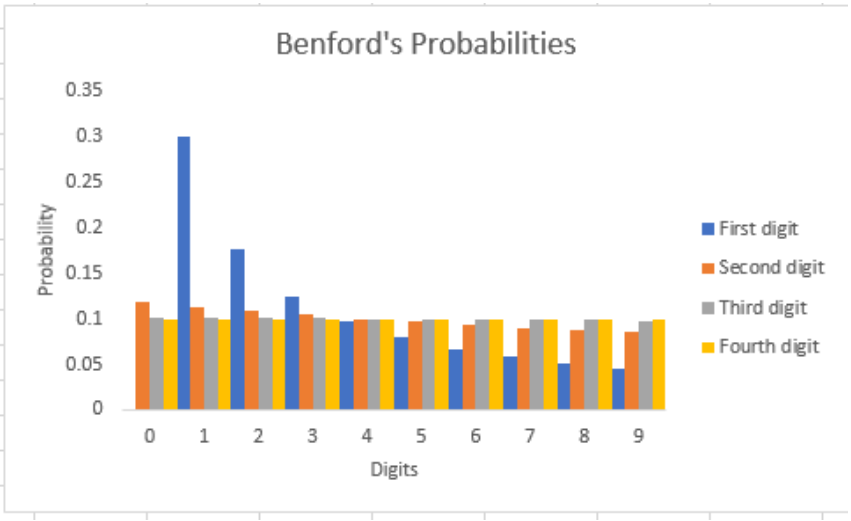
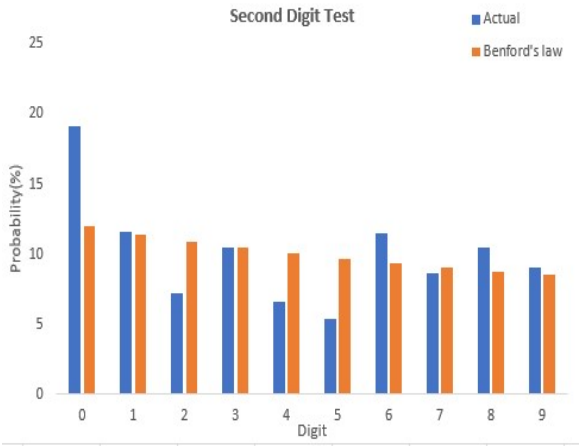


Figure 7.2: BL probabilities for digits $d \in \{0, 1, 2, \ldots, 9\}$

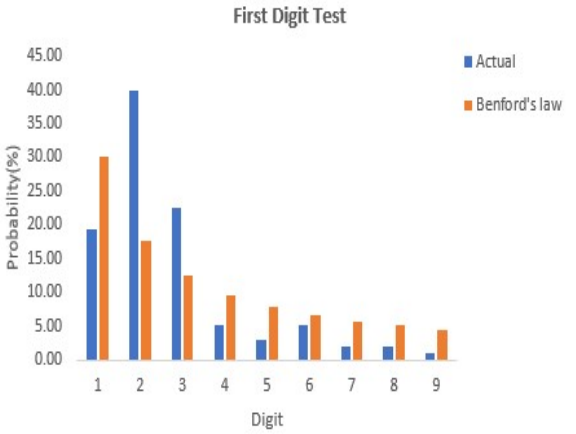
Five standard BL distribution tests are derived from Properties 1 and 2 above, with the simplest test being the FSLD. The applicability of these tests depends on the underlying numerical data. For example, if a feature has two digits, for example, 42, this implies that a test such as the third digit test will not be applicable. The BL distribution tests are described below:

- The first digit test (FDT).
- The second digit test (SDT).
- The first two digits test (F2DT).
- The third digit test (TDT).
- The last two digits test (L2DT).

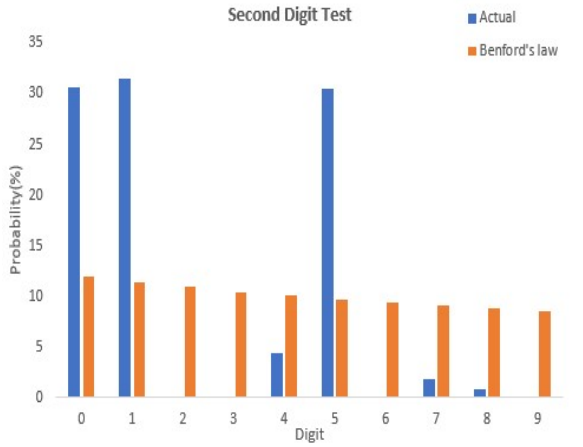
The example that follows illustrates the five BL distribution tests using the "total length of backward packet" feature from the CICDDOS2019 NIDSs cybersecurity data set. In this example, BL distributions are compared with the distributions of the leading digit of the actual data. Furthermore, this example illustrates how BL can be adopted as a feature selection method.



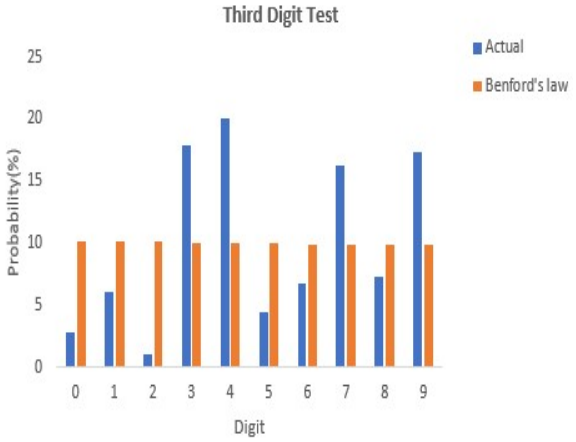
(a) Benign SDT



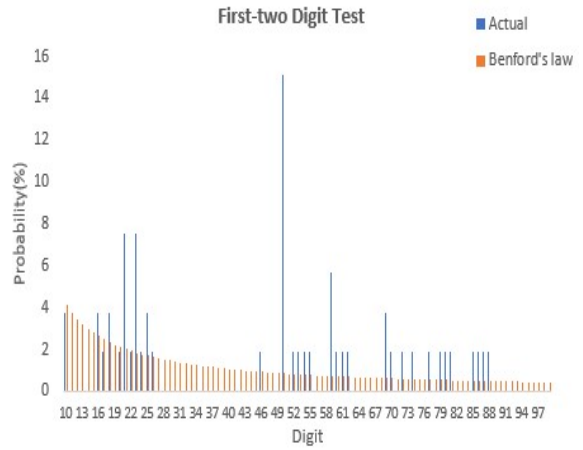
(b) DDoS FDT



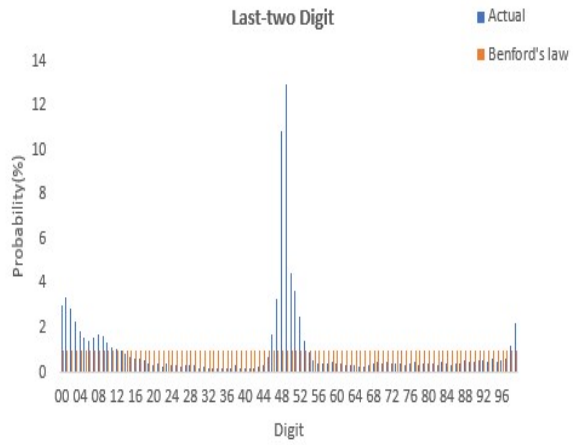
(c) DDoS SDT



(d) DDoS TDT



(e) DDoS F2DT



(f) DDoS L2DT

Figure 7.3: Comparing BL and actual distributions for total length of backward packet feature

Figure 7.3 illustrates BL and actual distributions for the total length of backward packet feature in benign and DDoS DNS network traffic. This feature closely follows the SDT of BL on benign network traffic, whereas the same feature does not follow the FDT, SDT, TDT, F2DT,

and L2DT on malicious DDoS DNS network traffic. Since this feature is able to distinguish between benign and DDoS DNS network traffic, it can be considered a significant feature in this context.

Not all data sets are expected to conform to BL [282, 288, 291], and in the next section, the minimum conditions of a BL data set are stated.

7.2.2.5 Benford's law - data set prerequisites

This section stipulates the minimum data set conditions that must be satisfied in order to apply BL distributions.

- *Leading digits from 1 to 9 must all be possible in a data set [292, 288, 282]* : Consider the example in Figure 7.3 whereby leading digits 1 to 9 are all possible.
- *A data set must occur naturally [13, 293]* : This simply means a real-world data set such as OSN and NIDS data sets, that has natural numbers.
- *Numbers should not be sequential [282, 13]* : For example, numbers for the total length of backward packet feature, e.g., Figure 7.3 occur randomly and not in a sequential order.
- *Numbers should not have pre-defined minimum and maximum values [282, 13]* : For example, a Boolean type of feature such as the "is_sm_ips_ports", which compares source and destination IP addresses in NIDSs and assigns 0 or 1 values, will violate this requirement. The majority of features found on OSNs and NIDSs do not have pre-defined minimum or maximum values [189, 260].
- *A data set must be large enough (usually one thousand sample size) [219, 182]* : OSNs and NIDSs have large sample sizes [189, 260].

The next section provides details of why BL was considered a suitable feature selection method in this thesis.

7.2.2.6 Motivation for using Benford's law as a feature selection method

BL was considered a feature selection method, based on the following reasons:

- BL has interesting properties such as the expected distributions of leading digits on data sets, and these properties can be used to identify anomalous behaviours on cybersecurity data sets that are considered as big data [219, 182]. For example, every number has a unique leading digit that span the $\{1, 2, \cdots, 9\}$ dimension, hence, one can effectively compare distributions of leading digits despite differences in the magnitude of numbers.
- BL does not require any class balance of data sets prior to implementation [260, 294], and does not require any parameter fitting [282]. OSNs and NIDSs are considered big data platforms as mentioned previously, therefore, implementing class balance and parameter fitting will be a daunting task on big data platforms [37, 241].
- BL is computationally efficient on high-dimensional big data platforms such as OSNs and NIDSs [189, 295], given the aforementioned point about parameter fitting.
- BL is based on simple equations which are intuitive, as opposed to methods such as principal component analysis (PCA), which require a strong mathematical background understanding [189].
- BL is scale invariant [285] meaning that the unit of measurement of features on cybersecurity data sets does not have a negative impact for the purposes of feature selection. For example, measuring flow_duration feature in seconds or minutes, means the same thing in the context of BL.

The overall distance between the BL and observed distributions is measured using a conformity test to determine whether the observed distributions follow BL distribution or not.

7.2.3 Conformity tests

Conformity tests quantify the distance between observed and expected distributions [296]. Among the many conformity tests, the author of this thesis chose to use the Pearson chi-squared test and the MannWhitney U test in this thesis to compute the p-value. In all conformity tests,

the goodness-of-fit test is formulated as follows:

$$\text{Null hypothesis } (H_0) = \text{a distribution obeys BL} \tag{7.3}$$

$$\text{Alternative hypothesis } (H_1) = \text{a distribution violates BL} \tag{7.4}$$

If p-value < 0.05, H_0 is rejected; otherwise, H_0 cannot be rejected. Consider Figure 7.3, where the total length of backward packet feature obeys the SDT of BL on the benign network traffic; thus H_0 cannot be rejected. However, BL distributions are violated on the DDoS_DNS malicious network traffic, as p-value < 0.05 and H_0 is rejected.

7.2.3.1 Pearson chi-squared

The Pearson chi-squared test sums the differences between the actual observed data, i.e., feature data and the expected model, known as the BL distribution. The actual observed data is assumed to be obtained from a random sample, and the sample size must be sufficiently large. This equation is formulated as follows:

$$\chi^2 = \sum_{i=1}^{N} \frac{(o_i - e_i)^2}{e_i} \tag{7.5}$$

where o_i and e_i are observed and expected digits for $i = 1, \cdots, N$.

7.2.3.2 Mann-Whitney U test

The MannWhitney U test states that given samples of sizes n_1 and n_2 (e.g., two feature data sets) with rankings R_1 and R_2 respectively, compute

$$U_1 = n_1 n_2 + \frac{n_1(n_1+1)}{2} - R_1 \tag{7.6}$$

$$U_2 = n_1 n_2 + \frac{n_2(n_2+1)}{2} - R_2 \tag{7.7}$$

$$U = \min(U_1, U_2) \tag{7.8}$$

For the purpose of the research at hand, once the significant features have been identified by BL on OSNs and NIDSs, they are used as inputs into machine learning algorithms to intelligently discover cyberattacks launched by bots. Machine learning (ML) is discussed next.

7.2.4 Machine learning

7.2.4.1 Introduction

Machine learning (ML) can be defined as a process that uses a sample data set to gain insight into and build intelligent decision-making models [35, 297, 298]. For example, consider a computer program that observes you labelling bot 𝕏 accounts as benign or malicious, and based on this knowledge, it learns how to better identify benign and malicious bot accounts. In this example - based on Mitchell's classical definition of an ML problem [299] - the task T is to label or classify 𝕏 bot accounts as benign or malicious, the experience E is the program observing you labelling the accounts, and the performance P is the number of bot accounts that are correctly classified as benign or malicious.

ML has been applied in various fields such as financial fraud detection [289, 280] and identity deception on OSNs [297]. A binary classifier is the simplest categorical classifier because it predicts one option from among two. If an ML algorithm predicts one option from among many (more than two), this type of ML problem is known as multi-classification [300]. In this thesis, the author of this thesis limited the focus to binary classifications as the aim was to classify bot activities on OSNs and NIDSs as benign or malicious. Depending on the availability of a labelled data set to train an ML algorithm, there are three main types of ML-based on supervised, unsupervised, and semi-supervised learning. In summary, a supervised ML algorithm is trained using a fully labelled data set; an unsupervised ML algorithm is trained using an unlabelled data set, and a semi-supervised ML algorithm is trained using a small sample of labelled and large samples of unlabelled data [301, 236, 57].

A common application of ML is anomaly or outlier detection [302, 76]. From a cybersecurity perspective, anomaly detection refers to the discovery of outliers in cybersecurity data sets [62, 63]. In general, the discovery or detection of anomalies in cybersecurity data sets leads to the identification of cyberattacks [202, 5]. For example, consider a Web login cybersecurity data set that indicates abnormal failed login attempts on a website, could indicate a brute-force attack [8]. Anomaly detection in discussed next.

7.2.4.2 Anomaly detection

Anomaly detection is a sub-field of ML that detects rare or unexpected data points that do not conform to a well-defined notion of expected behaviour [303, 304, 305]. Anomaly detection is typically applied to problems with high-dimensional imbalanced data sets; that is "normal" or expected data points are a significant majority group, while "anomalous" data points are a minority group [35]. This type of problem domain has several applications including credit card fraud and network intrusion detection. From a cybersecurity perspective, detecting or discovering anomalies is vital, as these anomalies often lead to the discovery of cybercrimes such as DDoS attacks [306, 307]. For example, consider an 𝕏 account that posts content frequently; this behaviour can be deemed anomalous [307, 308] even though the actual post content can be malicious (e.g., spam) or benign (e.g., news update). The current thesis focuses on discovering anomalous activities of bots in OSNs and NIDSs. Anomaly detection has been studied for many years [35], and over this period, several algorithms that include support vector machine (SVM) have been developed for different domains including OSNs and NIDSs. The underlying assumption in anomaly detection algorithms is that normal behaviour can be well formulated, such that data points deviating from this expected behaviour are deemed as anomalies [35, 307]. Next, supervised anomaly detection approaches are discussed.

Supervised anomaly detection approaches

Supervised anomaly detection algorithms assume that there are a sufficient number of labelled data points of normal and anomalous cases in the training data set [309, 308, 310]. This method works well for detecting known anomalies, as in the case of credit card fraud [76]. The most commonly used supervised algorithms include neural networks (NN) and SVM [304]. These algorithms can be challenged by class-imbalanced training data sets in which anomalous cases are far fewer than normal cases [311]. The issue of an imbalanced data set can be addressed by techniques such as random over-sampling or under-sampling [301]. A practical limitation of supervised anomaly detection approaches involves obtaining of accurate samples of labelled anomaly cases [309]. Furthermore, they are ineffective in discovering unknown anomalies such as zero-day attacks or variants of known anomalies [177]. Therefore, the supervised anomaly detection approach was not deemed suitable for solving the current research problem. Unsupervised anomaly detection approaches are discussed next.

Unsupervised anomaly detection approaches

Unsupervised anomaly detection algorithms implicitly assume that the majority of data points are normal cases, and these will be grouped around the same point, whereas anomalies will deviate from this normal group [307, 275]. The major advantage of unsupervised approaches is that they do not require labelled training data for either the normal or anomalous data points. If this assumption is not met, unsupervised algorithms can suffer from high false-positive, i.e., benign behaviour being incorrectly classified as malicious and false-negative, i.e., attacks that are undetected [174, 311]. Although algorithms such as isolation forests and distribution-based methods are popular for solving such problems [174], the author of this thesis decided against this approach, given that a sample of labelled data was obtained for discovering bot cyber threats in OSNs and NIDSs. Reinforcement anomaly detection approaches are discussed next.

Reinforcement anomaly detection approaches

Reinforcement anomaly detection algorithms trains a software agent to make informed decisions in a particular environment [312, 313]. The software agent continuously receives feedback from the environment to automatically improve its efficiency. The software agent is rewarded for making correct decisions and penalised for incorrect decisions [312, 313]. For example, if the software agent correctly discovers a malicious bot from $\mathbb{X}$, it is rewarded, and this information is used by the agent to improve its prediction. In this thesis, given that the CySecML methodology was not implemented in a real-world environment, reinforcement ML is disregarded. Semi-supervised anomaly detection approaches are discussed next.

Semi-supervised anomaly detection approaches

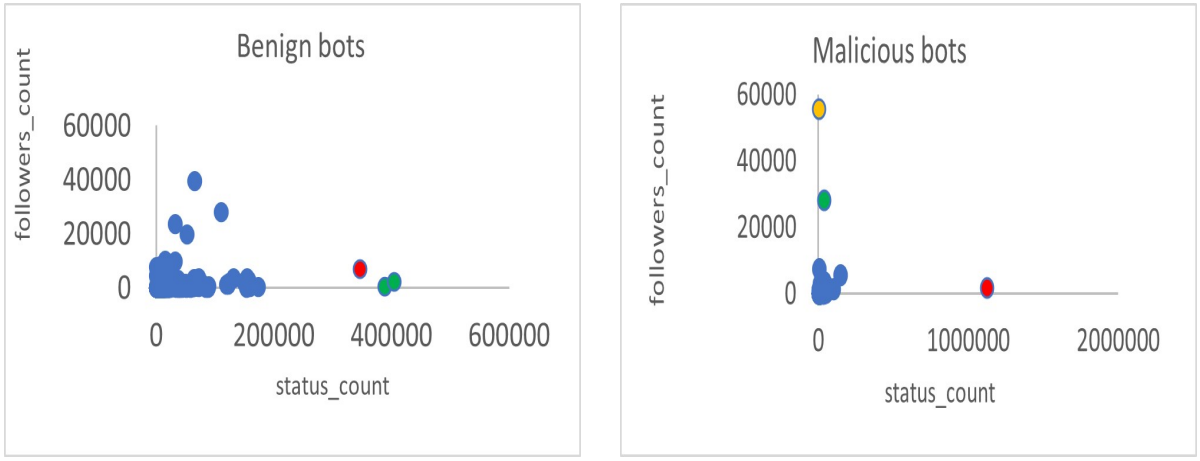
Semi-supervised anomaly detection algorithms fall between the supervised and unsupervised learning algorithms [73, 314]. These algorithms assume that the training data set consists of small labelled and large amounts of unlabelled data sets [304, 315]. Semi-supervised algorithms are applicable to most real-life problems including cybersecurity problems such as discovering bot malicious activities [35, 303]. For the case at hand, the author of this thesis obtained a sample of labelled data and large samples of unlabelled data for cyberattacks discovered on OSNs and NIDSs. Therefore, this thesis adopted semi-supervised ML, further on referred to as SSML.

Inductive and transductive learning are the two main SSML approaches [236, 298, 316]. Recall that in a SSML setting, the training data set consists of unlabelled data points [236, 316], and therefore, two distinct learning objectives can emerge from this. One is to build a classifier and predict labels of unseen test data - a learning approach known as inductive SSML. The other learning goal is to predict labels of the unlabelled data seen in the training phase - an approach known as transductive SSML [73, 316]. A common analogy used to differentiate between these two SSML approaches is that transductive learning is an exam test that is based on questions a student has encountered before, whereas inductive learning involves new questions [316]. In the current thesis, inductive SSML was adopted as the objective of this thesis was to build a classifier that can discover cyber threats launched by bots.

A plethora of SSML algorithms exists, each with its own advantages and disadvantages, see [73, 236, 301]. For the purposes of this thesis, the author of this thesis sampled various types of inductive SSML algorithms that have been used in big data platforms such as OSNs and NIDSs. The types of inductive SSML algorithm included graph-based algorithms, maximisation separation-based algorithms and generative-based algorithms. The graph-based algorithms considered were label propagation (LP) and label spreading (LS). The maximisation separation-based algorithms considered were the one-class support vector machines (OCSVM) and semi-supervised support vector machines (S3VM). The generative-based algorithm was the Gaussian mixture model (GMM). There are two main assumptions for SSML approaches:

- Continuity or smoothness assumption: Data points that are closer to each other are likely to share the same output label [236].
- Cluster assumption: Data points in the same cluster are likely to share the same output label [236].

Figure 7.4 demonstrates continuity and cluster assumption using data from $\mathbb{X}$.



(a) 𝕏 - benign bots

(b) 𝕏 - malicious bots

Figure 7.4: Benign and malicious 𝕏 bots using data from [4]

Figure 7.4 is based on a data set from Oentaryo [4] that was used in this thesis. It illustrates benign and malicious bots by plotting "status_count" and "followers_count" features. The data points coloured in blue are "closer" to each other; thus, they will have the same label, by continuity and cluster assumption as compared to non-blue data points i.e., outliers.

The SSML algorithms used in this thesis are briefly described below.

Brief introduction to the GMM algorithm

The Gaussian mixture model (GMM) aims to discover anomalies in a data set using the sum of weighted Gaussian distributions, which was established in the 1880s by German mathematician Carl Friedrich Gauss [317]. The GMM algorithm uses a training data set that is labelled to define "normal" or expected and "anomalous" or unexpected regions, such that a new (unlabelled) data point has a probability (ℙ) of belonging to either the normal or anomalous region [317].

Motivation for using the GMM algorithm for the InternetBotDetector model

The GMM algorithm has been widely used to discover anomalous behaviours in big data platforms such as OSNs and NIDSs [318, 236, 319]. Using NIDS data sets as an example, the GMM algorithm is first trained to define regions of normal and anomalous network traffic based on a particular set of features such as flow duration, packets and so on. Then, given a new unknown

network traffic point, the GMM algorithm determines whether this point belongs to the normal or anomalous network traffic region, given a probability $\mathbb{P}$. Specifically, the GMM algorithm assumes that normal data points are clustered around the same point in high-probability regions, whereas anomalies lie in low-probability regions [236].

The GMM algorithm was considered suitable for the study at hand as it can be used to discover anomalies on big data platforms such as OSNs and NIDSs. In addition, the GMM algorithm works for binary classification problems where a training data set contains unlabelled data points. The semi-supervised support vector machine (S3VM) algorithm is presented next.

Brief introduction to the S3VM algorithm

The S3VM algorithm was extended from the standard support vector machine (SVM) by [318] to solve machine learning (ML) problems when a training data set contains unlabelled data points. In contrast to the standard SVM that is based on supervised learning, i.e., the training data set contains only labelled data points, S3VM is based on semi-supervised learning that was originally proposed by Bennet and Demiriz in 2017 [318]. The core idea behind S3VM is to improve the generalisation performance of a standard SVM by using small amounts of labelled data points and a large number of unlabelled data points for training a classification model [73, 318].

Motivation for using the S3VM algorithm for the InternetBotDetector model

The S3VM algorithm has been proposed by various authors [318, 320, 321] to discover anomalies on big data platforms. For example, Pan et al. [321] proposed using the S3VM algorithm to train a classification model for discovering anomalies on an unmanned aerial vehicles (UAVs) platform. The S3VM algorithm was also deemed suitable for the study at hand, given its ability to solve binary classification problems and discover anomalies where a training data set contains unlabelled samples. The label propagation (LP) algorithm is presented next.

Brief introduction to the LP algorithm

The label propagation (LP) algorithm is a graph-based SSML that stems from the assumption that data points that are clustered will have the same label, whereas data points away from this cluster will have different labels [236]. Given N labelled data points, i.e., $y = \{+1, -1\}$ and M unlabelled data points, i.e., $y = 0$. Let $G = \{V, E\}$ represent a graph with vertex E consisting of labels $V = \{+1, -1, 0\}$ where the edge is based on the affinity matrix W [236]. The latter, also known as the similarity matrix, which is a statistical method that measures pairwise similarities between the data points. For instance, if two data points belong to the same cluster, they should be highly related, in other words, they should have the same label. Let input features be denoted by a vector X. The LP algorithm assumes that two nodes are connected if they are "similar"; therefore, unlabelled data points can be labelled by propagating the labelled data points until convergence is achieved [301, 236].

Motivation for using the LP algorithm for the InternetBotDetector model

The LP algorithm is a popular graph-based SSML algorithm, according to [236, 316, 73]. Its popularity stems mainly from its flexibility to work as both a transductive and an inductive SSML [322, 73]. Using a OSN data set as an example, the LP algorithm uses small labelled bot data points that are benign (+1) and malicious (-1). It will cluster these labelled data points according to their similarity, and this knowledge is then used to label the unlabelled data points according to their clusters [322, 316]. To summarise - the LP algorithm uses both labelled and unlabelled data points to build a binary classification problem to discover anomalies in big data platforms [322, 316]. The label spreading (LS) algorithm is introduced next.

Brief introduction to the LS algorithm

The label spreading (LS) algorithm is similar to label propagation (LP), except that the labels of vertex $G = \{V, E\}$ may update during the iteration process [236]. Specifically, the clamping factor $\alpha \in (0, 1]$ determines whether the labelled data point will update or not. If $\alpha = 0$, this implies that the original label remains the same, and thus the LS method will behave the same way as the LP algorithm.

Motivation for using the LS algorithm for the InternetBotDetector model

The LS algorithm is similar to the LP algorithm with the exception that LS can change the labels of data points [73, 316], e.g., a labelled data point may have label +1 on iteration 0 and label -1 on iteration 1. Hence, the author of the thesis decided to include the LS algorithm in this thesis to assess how it performs in comparison to the LP algorithm with regard to the problem at hand. The one-class support vector machines (OCSVM) algorithm is introduced next.

Brief introduction to the OCSVM algorithm

The one-class support vector machines (OCSVM) algorithm as extended from the standard SVM algorithm was originally introduced by Vapnik et al. [300, 236]. The objective of the OCSVM is to train a classifier using only one class, e.g., benign network traffic, such that it can learn to classify anomalies, i.e., malicious network traffic [323]. In this context, one class does not imply identical data; however, it refers to the same label of different data points.

Motivation for using the OCSVM algorithm for the InternetBotDetector model

The OCSVM algorithm is an extension of the standard SVM algorithm that aims to discover anomalies in data sets [323, 215, 300]. The OCSVM is a type of SSML [175] that is trained using one class of data points to define normal or expected behaviours, such that a new unlabelled data point can be labelled as expected or unexpected (anomalous) [323, 300]. Therefore, the OCSVM algorithm was deemed suitable for the study at hand, which aims to solve a binary classification problem. The OCSVM algorithm has been widely used by various authors to discover anomalies in OSNs [215] and NIDSs [300].

The challenges and limitations facing SSML algorithms in discovering anomalies in high-dimensional big data platforms such as OSNs and NIDSs are discussed next.

Challenges in discovering bot cyberattacks using machine learning models

As mentioned in the previous section, this thesis adopted the SSML anomaly detection approach to intelligently discover cyberattacks launched by bots on OSNs and NIDSs. Anomaly detec-

tion models use raw data to define a domain of normal data points such that any data points deviating from this normal domain are declared as anomalies. The above process can be challenging based on various factors - some of which are applicable to OSNs and NIDSs - as listed below.

• The presence of high-dimensional imbalanced big data, characterised by large volumes, velocity and variety, makes it difficult for any anomaly detection algorithm to accurately account for every possible normal data point in defining a normal region. Consequently, data points that lie on the edge of a boundary of the normal region may be incorrectly classified as anomalies, or vice versa [73, 316]. For example, it is not a straightforward task to accurately define a normal region for closely related data sets such as benign and malicious bot data sets.

• On big data platforms such as OSNs and NIDSs, data sets constantly evolve at a very fast pace; thus, the notion of normal and anomalous behaviour may also be changing fast. Feature selection methods and anomaly detection algorithms must be adapted to deal with these evolving data sets. BL was shown in [189, 295] to be effective feature selection method on OSN and NIDS data sets

• The lack of correctly labelled data sets such as cyberattacks launched by bots for training anomaly detection algorithms is often a challenge, which can cause a machine learning algorithms to be prone to high false-positive rates of anomaly detection algorithms.

The next section provides experiments conducted in relation to feature selection using BL used in this thesis.

7.3 Experimental work

7.3.1 Feature selection on NIDSs cybersecurity data sets

As part of feature selection, the next steps are to compute BL distributions, actual distributions, conformity test and identify significant features as depicted in Figure 7.1. Feature selection for NIDS cybersecurity data sets, the author of this thesis created a zero-day type network intrusion

attack from existing attacks in different data sets. Specifically, using "Approach 2" of creating *known unknowns*, the original data set such as the IoT intrusion data set in Table 5.1 is divided into two groups: benign and zero-day network traffic types. For instance, the IoT intrusion data set in Table 5.1 consists of benign and eight malicious network traffic types of attack. In this case, a new unknown zero-day attack type is created by combining the eight malicious network traffic types across the same features in Appendix C.1. At this stage, the question is whether there are any significant features that can differentiate between these two groups, i.e., benign and zero-day network traffic types.

Computing BL and actual distributions

The hypothesis in this regard is that once an attack has been launched, network traffic data of an attack will differ from normal benign network traffic data [175, 178]. Therefore, in this thesis, features that obey one of BL distributions (see Figure 7.2) on the benign network traffic and simultaneously violate all BL distributions on the zero-day network traffic are deemed to be significant [260]. Features that do not satisfy this condition are deemed redundant and are not used as inputs into an ML algorithm as they fail to differentiate between benign and zero-day network traffic. As a benchmark, the significant features identified by the BL method on each NIDS data set are compared with those identified by the original authors of each NIDS data set. In each of the data sets considered in this thesis, BL distribution is compared with the actual distribution using goodness-of-fit measures to quantify the difference between the observed and expected distributions. The Pearson chi-squared goodness-of-fit used in this thesis was chosen from a plethora of goodness-of-fit measures. The goodness-of-fit test was formulated in Section 7.2.3.

BL expected leading distributions are computed using equations 7.1 and 7.2. The actual leading distributions are computed by first rounding input data (i.e., feature values) to the nearest whole numbers and counting the actual observations of leading digits 1 to 9. Using the cybersecurity data sets presented in Chapter 5, this section demonstrates the effectiveness of BL as a feature selection method. The significant features of the IoT Intrusion-2020 data set are discussed next.

7.3.1.1 IoT Intrusion-2020 feature selection results

Ullah et al. [8] identified significant features in the IoT Intrusion-2020 network intrusion data set, see Appendix C.1 using the recursive feature elimination (RFE) method. Table 7.2 summarises the significant features identified by BL method on the IoT Intrusion-2020 data set.

	Benford's law (BL) vs Ullah et al. [8]
BL	8,15,16,17,18,19,21,22,23,24,25,26,28,43,46,47,48,49,59,60,61,68,69,70,71,74
[8]	3,5,6,13,14,15,17,18,19,37,39,45,46,47,54,58,59,60,61,62,63,64,65,66,68,69,72,73,75

Table 7.2: Summary of significant features identified on the IoT Intrusion-2020 data set by Ullah et al. [8] and the BL method

In Table 7.2, 26 significant features were identified by the BL method, whereas Ullah et al. [8] identified 29 from the data set presented in Appendix C.1. Of the 29 significant features identified by Ullah et al. [8], 11 overlapped with those identified using the BL method. The latter method does not perform well on binary-type features, such as ACK_Flag_Count (54), Forward_PSH_Flags (37), and features that do not span the $\{1, 2, \cdots, 9\}$ dimension. In summary, the BL method identified features such as flow duration, packets, and byte features to be significant for differentiating between benign and zero-day network traffic. The significant features of the CICDDOS2019 data set are indicated next.

7.3.1.2 CICDDOS2019 feature selection results

Sharafaldin et al. [9] identified significant features in the CICDDOS2019 network intrusion data set, see Appendix C.3 using a random forest regressor (RFR) feature selection method.

	Benford's law (BL) vs Sharafaldin et al. [9]
BL	8,12,14,15,16,18,19,23,24,25,26,27,28,30,33,35,45,46,47,48,49,59,71,73
[9]	5,6,8,11,13,14,16,23,25,26,27,28,30,41,43,45,46,48,54,59,62,70,73,76

Table 7.3: Summary of significant features identified on the CICDDOS2019 data set by Sharafaldin et al. [9] and the BL method

Table 7.3 shows that the BL method and Sharafaldin et al. [9] identified 24 significant fea-

tures. Of the 24 significant features identified by Sharafaldin et al. [9], 14 overlapped with those identified by means of the BL method. A similar finding is observed here as in the IoT Intrusion-2020 data set relating to binary-type features, and features that do not span the $\{1, 2, \cdots, 9\}$ dimension. BL identified features such as packets and flow duration to be significant for differentiating between benign and zero-day network traffic. The significant features of the UNSW-NB15 data set are indicated next.

7.3.1.3 UNSW-NB15 feature selection results

Moustafa et al. [2] used the association rule mining (ARM) feature selection method to identify the significant features of the UNSW-NB15 data set, see Appendix C.2.

	Benford's law (BL) vs Moustafa et al. [2]
BL	5,6,7,12,13,15,16,19,21,22,27,31,33,36,40,41
[2]	5,6,8,10,11,12,13,14,15,19,20,21,23,25,26,27,31,32,33,34,35,36,37,38,40,41,42

Table 7.4: Summary of significant features identified on the UNSW-NB15 data set by Moustafa et al. [2] and the BL method

Table 7.4 shows that the BL method identified 16 significant features, whereas Moustafa et al. [2] identified 27. Of the 27 significant features identified by Moustafa et al. [2], 13 overlapped with those that were identified using the BL method. Similarly, BL does not perform well on binary features such as is_ftp_login (37) and is_sm_ips_ports (42) and features that do not span the $\{1, 2, \cdots, 9\}$ dimension. The BL method deemed source-to-destination, destination-to-source packets, and packet size information important for differentiating between benign and zero-day network intrusion attacks. The significant features of the CIRACICDOHBRW2020 data set are discussed next.

7.3.1.4 CIRACICDOHBRW2020 feature selection results

Montazerishatoori et al. [10] identified significant features in the CIRACICDOHBRW2020 network intrusion data set, see Appendix C.4 using a random forest regressor (RFR) feature selection method.

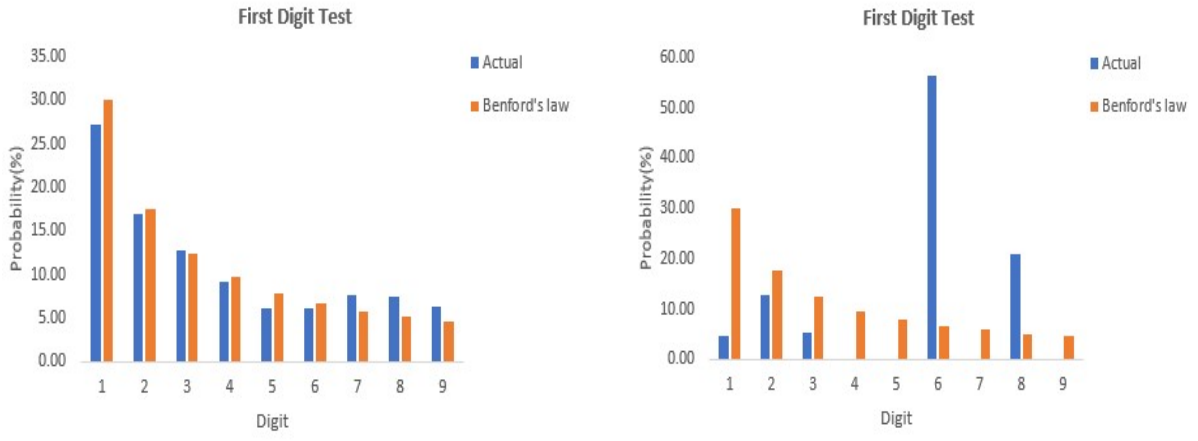
	Benford's law (BL) vs Montazerishatoori et al. [10]
BL	6,7,8,9,10,11,12,13,14,15,19,20,21,22,23,27,28,34
[10]	7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34

Table 7.5: Summary of significant features identified on the CIRACICDOHBRW2020 data set by Montazerishatoori et al. [10] and the BL method

Table 7.5 shows that BL identified 18 significant features while Montazerishatoori et al. [10] identified 28 significant features. Of the 28 significant features identified by Montazerishatoori et al. [10], 17 overlapped with those identified by the BL method. The BL method identified flow and packet features to be significant for differentiating between benign and malicious zero-day network traffic. However, the BL method did not perform well on features such as PacketLengthCoefficientofVariation (18) that do not span the $\{1, 2, \cdots, 9\}$ dimension.

7.3.1.5 Summary - significant features for the discovery of zero-day network intrusion attack types

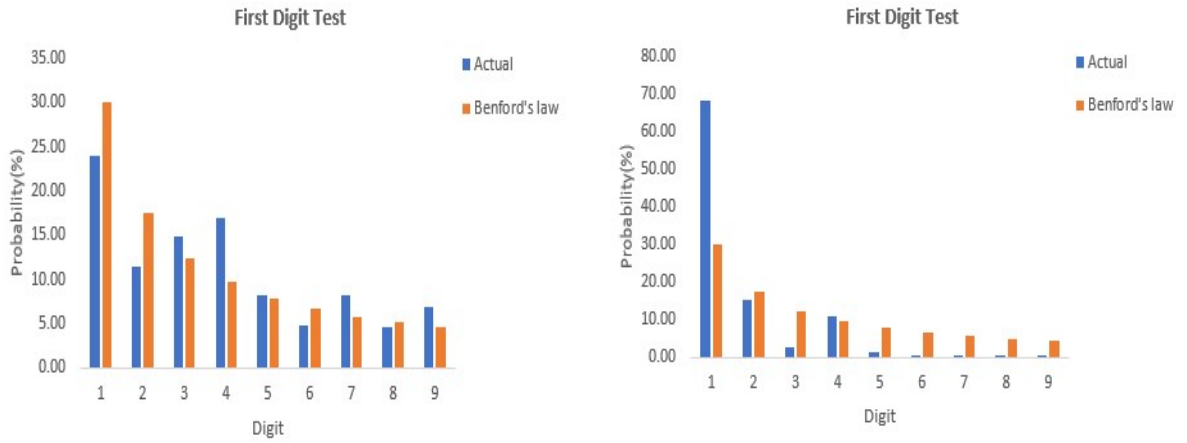
The figures below illustrates the BL distribution for each data set for benign and zero-day network traffic. In this regard, benign network traffic obeys the BL distribution, whereas zero-day malicious network traffic violates it. More detailed results are provided in Appendix D.



(a) Benign network traffic

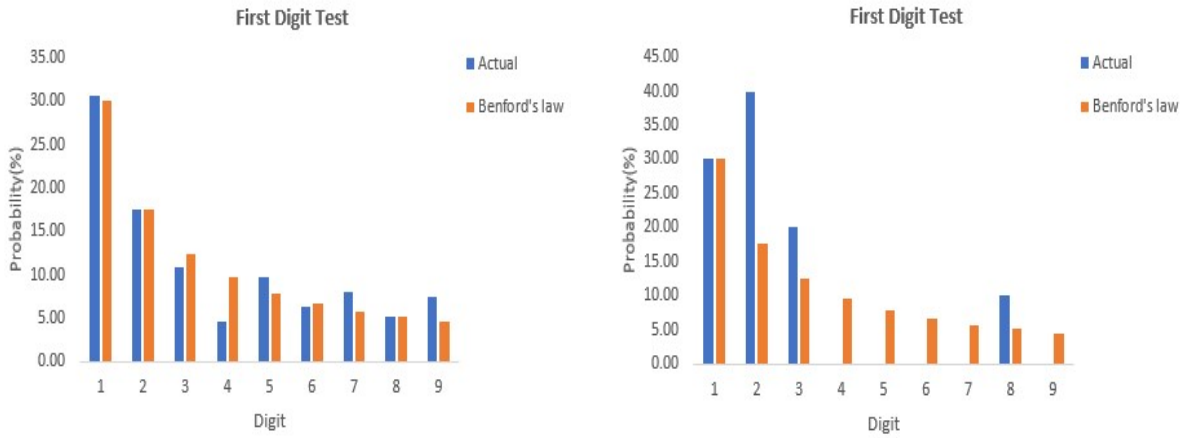
(b) Malicious zero-day network traffic

Figure 7.5: Comparison between the FDT benign and zero-day network traffic on the UNSW-NB15 data set



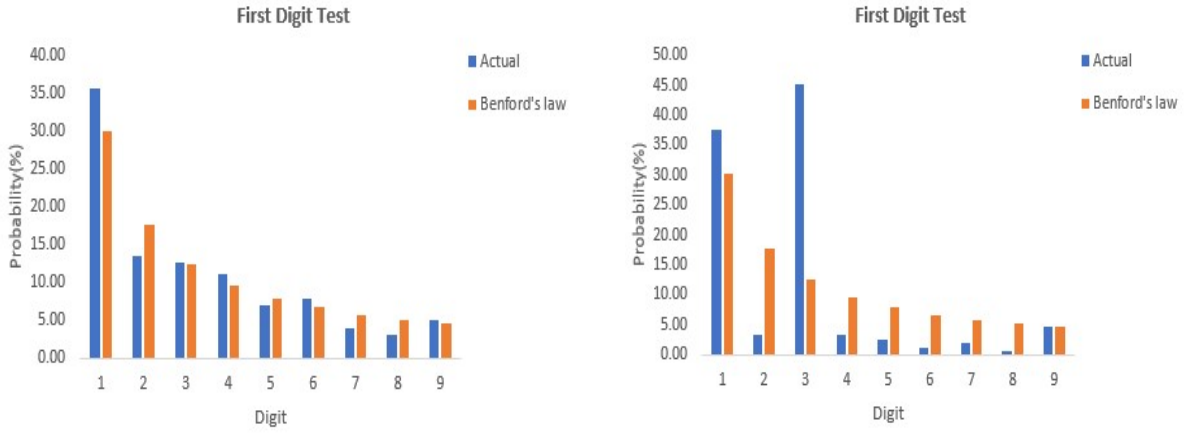
(a) Benign network traffic (b) Malicious zero-day network traffic

Figure 7.6: Comparison between the FDT benign and zero-day network traffic on the CICD-DOS2019 data set



(a) Benign network traffic (b) Malicious zero-day network traffic

Figure 7.7: Comparison between the FDT benign and zero-day network traffic on the IoT Intrusion-2020 data set



(a) Benign network traffic

(b) Malicious zero-day network traffic

Figure 7.8: Comparison between the FDT benign and zero-day network traffic on the CIRACICDO- HBRW2020 data set

In each of the NIDSs data sets above, the significant features for the discovery of zero-day network intrusion attack types are indicated in Tables 7.2, 7.3, 7.4 and 7.5 - based on BL method. Using these results, the author of this thesis identified the common set of significant features in all the data sets used and summarised them below:

- *Flow duration* defined as a measure of a set of packets passing a particular point in a network during a certain time interval [324, 325]. Flow features have been shown in the past to be effective for discovering malicious network traffic, see [324, 266, 175]. In this instance, zero-day network traffic was observed to have longer flow duration than benign network traffic.

- *Flow inter-arrival times* measure the time between two consecutive network flows. The flow inter-arrival times measure was adopted by [186, 279, 183] to discover anomalous network traffic types. Similar observations as in the flow duration case above.

- *Forward and backward packets* measure packets received and sent in a network [8]. In this instance, zero-day network traffic were observed to have a high number of forward and backward packets on average, compared to benign network traffic.

- *Packet sizes* are a measure of an actual packet size [8]. When attackers attempt to overload a network by sending large amounts of random data, the packet sizes of this data will be relatively large for malicious zero-day network traffic compared to benign network traffic.

- *Segment size* can be described as the largest amount of data in bytes that a device can handle [8]. Similar observations as in the packet sizes above.

- *Source to destination or destination to source packet count* measures the number of packets from source to destination, and vice versa [2]. Similar observations as in the packet sizes above.

- *Source/destination TCP base sequence number* can be described as a number used to keep track of every byte sent or received by a host [2]. In this instance, zero-day network traffic displayed a high number TCP base due to attackers attempting to overload a network with random data.

- *Number of connections of the same source/destination address* contains information about the number of connections of the same source/destination address [10]. This behaviour was mainly observed in DNS based botnet zero-day attacks because botmasters use C&C to communicate with botnets.

The significant features identified by the BL method discussed above for the intelligent discovery of zero-day network intrusion attacks are consistent with those in existing studies. For instance, Vahdani et al. [167] found that the duration and packet-based features, such as the size of the packet, are significant for discovering zero-day network intrusion attacks. Duong et al. [168] found connection-related features such as the number of connections to different servers to be significant for the discovery of zero-day network intrusion attacks. Zavrak et al. [175] found that flow-based and packet-based features are significant for the discovery of zero-day network intrusion attacks. Zoppi et al. [178] identified packet-based methods, such as the average size of a packet, to be significant in discovering zero-day network intrusion attacks.

The next section focuses on feature selection on OSN cybersecurity data sets.

7.3.2 Feature selection on OSN cybersecurity data sets

Feature selection on OSN cybersecurity data sets is presented in the next section for the purposes of intelligently discovering social engineering attacks.

7.3.2.1 Introduction

BL was implemented to identify significant features for differentiating between benign and malicious bots using a data set from Oentaryo et al. [4]. Furthermore, the BL results were benchmarked using ensemble random forest (ERF) and well-known feature selection methods from the ScikitLearn library [206]. The ScikitLearn library was chosen because it is one of the largest open-source libraries and it is mostly used by academics and practitioners [206].

7.3.2.2 Feature selection results using Benford's law (BL)

Given the number of digits in the features highlighted in Table 5.6, the author of this thesis decided that the first significant leading digit (FSLD) test is appropriate for identifying significant features that differentiate between benign and malicious bots. For example, the screen_name length feature has a maximum number of two digits, thus a TDT test cannot be applied. The same equations 7.3 and 7.4 were applied to identify the significant features.

Feature	Benign bots	Malicious bots
Favourites_count	**Cannot reject** H_0	**Reject** H_0
Lists_count	**Cannot reject** H_0	**Reject** H_0
Statuses_count	**Cannot reject** H_0	**Reject** H_0
Status.retweet_count	**Cannot reject** H_0	**Reject** H_0
Friends_count	**Cannot reject** H_0	**Reject** H_0
Followers_count	**Cannot reject** H_0	**Reject** H_0
Status.favorite_count	Reject H_0	Reject H_0
Screen_name length	Reject H_0	Reject H_0
Account age (in days)	Reject H_0	Reject H_0
URL_count	Reject H_0	Reject H_0
Status.reply_count	Reject H_0	Reject H_0
Hashtag_count	Reject H_0	Reject H_0

Table 7.6: BL feature selection results for benign and malicious bots using the FSLD test from 𝕏 data set

In Table 7.6, features in bold are significant because they obey BL on the benign bot data set but simultaneously violate BL on the malicious bot data set. If a feature violates or does not violate BL on either data set, then that feature is not deemed significant [189, 294]. Features such as screen_name length and account age (in days) violated BL for both benign and malicious bot data sets, and were deemed insignificant as they failed to differentiate between the two data sets and do not provide much insight into the behaviour of an account. On the other hand, features such as statuses_count, favourites_count, status.retweet_count, friends_count, followers_count, and lists_count differentiate between benign and malicious bot data sets. In this instance, known malicious bot behaviours, such as re-posting content more frequently than posting the original content, were observed.

In [189], the author of this thesis proposed BL as a feature selection method to differentiate between human and malicious bot 𝕏 accounts. Interestingly, it was observed that the significant features in Table 7.6 were also significant for differentiating between human and malicious bot accounts. This suggests that human and benign bot accounts display similar behaviour.

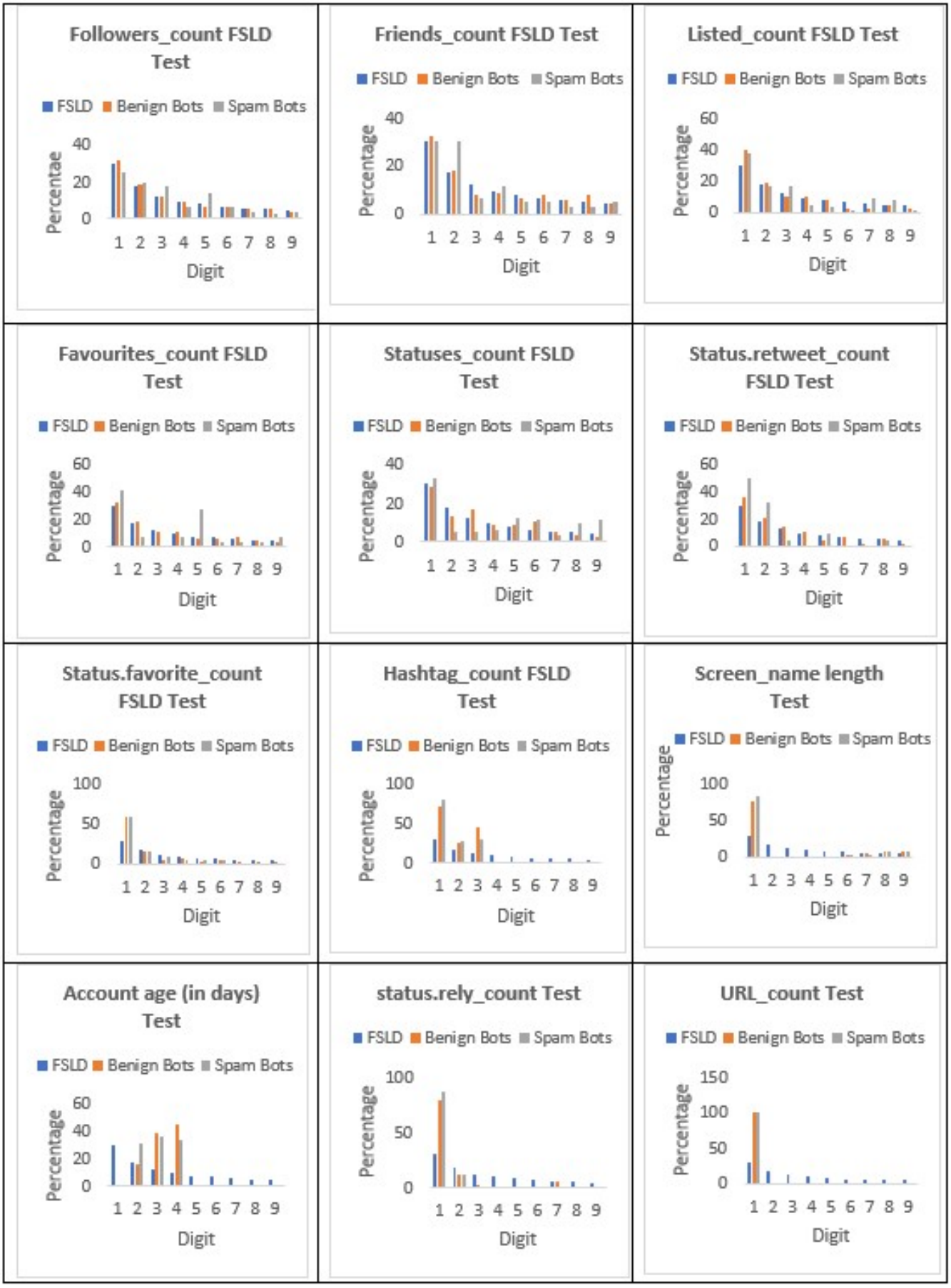


Figure 7.9: A sample of the FSLD Benford's law results in OSNs

Figure 7.9 depicts the BL feature selection results obtained using the FSLD test. For the features in bold (Table 7.6), benign bots closely followed BL, whereas malicious bots violated it.

7.3.2.3 Feature selection results using the ensemble random forest (ERF) method

The ensemble random forest (ERF) feature selection method is most used for high-dimensional imbalanced data problems, given the hierarchical structure that allows them to learn from the majority and minority classes [301]. In this thesis, benign and malicious bots were considered as the majority and minority classes respectively (see Table 5.5). The results in Figure 7.10 indicate that status_count, followers_count, friends_count, listed_count, favourites_count, and retweet_count are significant features for differentiating between benign and malicious bots. The bar length indicates the importance of each feature.

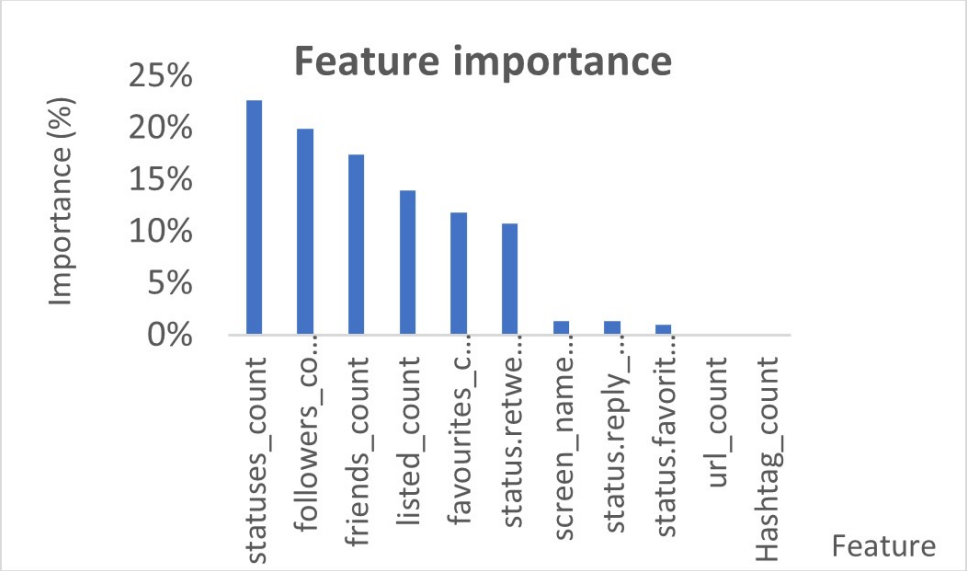


Figure 7.10: Feature importance using ERF for benign and malicious bots

Next, three additional feature selection methods from the ScikitLearn library were implemented to benchmark the BL feature selection results shown in Table 7.6.

Feature selection using false-positive rate (FPR)

The FPR method selects significant features from a data set, based on univariate statistical tests. This method selects significant features based on a p_value computed using the chi-squared test or analysis of variance, see [206] for more details. Using a p_value of 0.05, status_count, favourites_count, lists_count, friends_count, followers_count, and status.retweet_count were iden-

tified as significant features.

Feature selection using false discovery rate (FDR)

The FDR method identifies significant features using the Benjamini-Hochberg algorithm and an upper bound alpha on the expected FDR, see [206] for more details. The FDR method produced the same results as the FPR method, with an alpha value of 0.05.

Feature selection using familywise error (FWE)

The FWE method identifies significant features using the Bonferroni algorithm and an upper bound alpha, see [206] for further details. This method produced the same results as the FPR method, with an alpha value of 0.05. In summary, the significant features identified by BL to differentiate between benign and malicious bots were consistent with those identified by the ERF, FPR, FDR, and WE methods.

7.3.2.4 Summary - significant features for the discovery of bot social engineering attacks

Table 7.6 lists the significant features for the discovery of malicious bots in OSNs that launch attacks (also social engineering attacks). The significant features in this regard are favourites_count, lists_count, statuses_count, friends_count, followers_count, and status.retweet_count. The main purpose of a social engineering threat actor is to create an environment of false trust in order to manipulate people towards committing security mistakes [142, 326]. Malicious bots can manipulate the significant features by for example purchasing followers [137] to make their accounts appear trustworthy [45, 128]. These features have been shown by various authors [45, 4, 189] to be effective in discovering malicious bots in OSNs. The identified significant features are then used as inputs into a ML algorithm.

7.4 Chapter summary

This chapter was dedicated to identifying significant features that can be used as inputs into ML algorithms to intelligently discover cyberattacks such as zero-day intrusion attacks in NIDSs and social engineering attacks in OSNs. Benford's law was proposed as a feature selection method for both NIDS and OSN cybersecurity data sets. The significant features identified by

BL on the OSN data set were consistent with the ERF, FPR, FDR and FWE methods. Similarly, for NIDSs, BL results were consistent with those identified by the original authors of the data sets used. It was observed that BL was not able to discern binary-type features or features that did not span the $\{1, 2, \cdots, 9\}$ dimensions. Intuitively, it is difficult to distinguish unknown classes in a data set using binary-type features, thus, the BL method does not perform well in this regard. The identified significant features are then used as inputs into the InternetBotDetector model, as illustrated by the prototypes in the next chapter.

Part II

The CySecML methodology prototype implementation

Chapter 8

The CySecML methodology proof-of-concept

8.1 Introduction

Part I of this thesis dealt with the development of the CySecML methodology including its pseudo code representation. The next step is to implement this pseudo code in a practical manner. This chapter implements the experiments conducted by the author of this thesis to discover cyberattacks launched by bots. For illustrative purposes, the UNSW-NB15 NIDS and OSN cybersecurity data sets were used to intelligently discover zero-day intrusions and social engineering attacks respectively. The InternetBotDetector model component was evaluated using the standard machine learning valuation measures discussed next, followed by the experiments conducted to intelligently discover bot cyberattacks.

8.2 Related work and background

A methodology must be implemented and evaluated in a practical manner to determine whether it meets its objectives and usability, among other requirements [327, 165]. This is done through a prototype [328, 327, 329], which is a preliminary version of a complete solution system built and tested to achieve a particular objective [328, 327]. Jeffrey et al. [165] define a prototype as a procedure that facilitates the performance evaluation of a developed system or methodology.

For example, if one proposes a NIDS to discover network intrusion attacks using a particular methodology, then this methodology must be tested in terms of its limitations and performance in discovering network intrusion attacks, prior to being implemented in the real-world as a final solution. Although prototypes are not complete solutions, they are working versions of a complete solution that can be enhanced to achieve better performance and usability [301, 329]. There are a range of benefits to prototypes, mainly involving the ability to implement a variety of experiments and hypotheses. The valuation measures of ML-based prototypes are presented next.

8.2.1 InternetBotDetector model - valuation measures

The InternetBotDetector model performance in discovering cyber threats is evaluated using well-known machine learning (ML) measures, namely precision, recall, F_1 score and MCC score [330, 331]. For all the experiments conducted in this thesis, the data sets used were imbalanced and therefore, the accuracy measure was not used. According to [330, 331], this measure is known to suffer bias in binary classification problems that have imbalanced data sets, hence it was not used in this thesis. The valuation measures that were used are defined as follows:

- $Precision = \frac{TP}{TP+FP}$
- $Recall = \frac{TP}{TP+FN}$
- $F_1\ score = \frac{2}{\frac{1}{Recall}+\frac{1}{Precision}} = 2\frac{Precision*Recall}{Precision+Recall}$
- $MCC\ score = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$

where;

- TP - true positive prediction.
- TN - true negative prediction.
- FP - false positive prediction.
- FN - false negative prediction.

For binary classification problems in which the data set is imbalanced, improving one measure usually comes at the cost of the other measure [330, 331]. For example, to improve the precision, the threshold for recall would reduce. This type of relationship between precision and recall - referred to as the precision-recall trade-off - is described next.

8.2.1.1 Precision-recall trade-off

In this thesis, positive cases were treated as cyberattacks, while negative cases were treated as benign cases. Assuming that the negative cases are a majority class and positive cases are a minority in a data set, then, the precision-recall trade-off can be described as follows, according to [330, 331]:

- High precision and high recall: An ML algorithm correctly classifies both positive and negative cases.
- High precision and low recall: An ML algorithm cannot classify positive cases well, but when it does, the classification is accurate.
- Low precision and high recall: An ML algorithm classifies positive cases well, but it may classify negative cases incorrectly.
- Low precision and low recall: An ML algorithm poorly classifies both positive and negative cases.

From a cybersecurity perspective, this thesis is more concerned about classifying positive cases (i.e., cyberattacks) correctly, therefore, a high recall score would be a good indication of this. Recall measures how well an ML algorithm can predict positive cases among all the actual positive cases [330]. In addition, F_1 and MCC scores provide a more balanced view of the performance of an ML algorithm in terms of the correct classification of both positive and negative cases [330, 331]; thus, high scores for these measures would indicate a good performance of an ML algorithm. The experiments conducted in this thesis to discover zero-day network intrusion attacks in NIDSs are presented next.

8.3 Experimental work

8.3.1 Discovering zero-day network intrusion attacks in NIDSs

8.3.1.1 Experimental setup

This section describes the experiments that the author of this thesis conducted to discover zero-day network intrusion in NIDSs. To create a zero-day network intrusion attack, "Approach 2", discussed in the previous chapters, was adopted. The significant features identified by Benford's law differed from those of the authors of the data sets used in this thesis. Therefore, the author of this thesis conducted experiments to evaluate the impact of the features used as inputs in a machine learning algorithm. The experiments were performed as follows.

- Experiment 1 uses significant features identified by Benford's law as input into an SSML algorithm.
- Experiment 2 uses significant features identified by the original authors of each data set as input into an SSML algorithm.

For illustration purposes, the UNSW-NB15 NIDS data set is used to provide proof-of-concept of the CySecML methodology in discovering zero-day network intrusion attacks. In addition, one-class support vector machines (OCSVM) semi-supervised machine learning (SSML) is used to evaluate the InternetBotDetector component of the CySecML methodology. In both experiments, the identified significant features were used as inputs into SSML to train the model and build its knowledge. Subsequently, this learned knowledge was tested on a so-called testing data set to evaluate its knowledge. In the current study, a 20-80% train-test split was used (unless specified otherwise).

8.3.1.2 Block I - data extraction experiments

The UNSW-NB15 NIDS cybersecurity data set was downloaded from the Internet by the author of this thesis, see Table 5.2 and Figure 5.6. The UNSW-NB15 data set contains 44 features, as shown in Appendix C.2. The size of this data set has about instances of 175 341, which is

ten times the number of features in this data set; therefore, there was no need for additional synthetic samples. Zero-day network intrusion attacks were labelled as described in Figure 5.5.

8.3.1.3 Block II - data storage and quality checks experiments

The original UNSW-NB15 data set was downloaded as CSV files (in column vectors) and stored in a local drive. The UNSW-NB15 data set was examined and found to be of quality, as shown in Section 6.3.2; thus, data cleaning and feature selection were performed next.

8.3.1.4 Block III - data cleaning and feature selection experiments

No data cleaning was required on the UNSW-NB15 data set in this thesis. Benford's law data set prerequisites was performed as follows;

- *Leading digits from 1 to 9 must all be possible in a data set* : Using the "ROUND()" function that returns rounded numbers, digits from 1 to 9 were observable on the UNSW-NB15 data set.
- *A data set must occur naturally* : This data set was developed by Moustafa et al. [2] based on various botnet synthetic attacks, and contains natural numbers.
- *Numbers should not be sequential* : Numbers were observed by the author of this thesis, and they did not occur in sequential order, as shown in Figure 5.6.
- *Numbers should not have pre-defined minimum and maximum values [282, 13]* : The UNSW-NB15 data set does not contain pre-defined min or max values, except for the Boolean type of features such as "is_sm_ips_ports".
- *A data set must be large enough (usually one thousand sample size) [219, 182]* : The UNSW-NB15 data set had significantly more one thousand samples, as shown in Table 5.2.

The UNSW-NB15 NIDS data set satisfied Benford's law prerequisites; hence, Benford's law was selected as a suitable feature selection method. The significant features of UNSW-NB15 were identified using the BL method, as shown in Table 8.1 below.

Significant feature	Description
5	Source to destination packet count.
6	Destination to source packet count.
7	Source to destination transaction bytes.
12	Source bits per second.
13	Destination bits per second.
15	Destination packets retransmitted or dropped.
16	Source interpacket arrival time (mSec).
19	Destination jitter (mSec).
21	Source TCP base sequence number.
22	Destination TCP base sequence number.
27	Mean of the packet size transmitted by the src.
31	No. of connections that contain the same service and source address.
33	No. of connections of the same destination address and connections.
36	No. of connections of the same source and destination address.
40	No. of connections of the same source address and connections.
41	No. of connections that contain the same service and destination address.

Table 8.1: The UNSW-NB15 BL feature selection results

The identified significant features are then used as inputs into a ML algorithm.

8.3.1.5 Block IV - machine learning and decision-making experiments

For illustrative purposes, the OCSVM ML algorithm was used to evaluate the InternetBotDetector model. The below figure depicts the Python environment of the OCSVM ML algorithm.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.font_manager
from sklearn import svm
from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score, classification_report, confusion_matrix, mat

X_train = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\UNSW-NB15\X.csv")
X_train = X_train.replace((np.inf, -np.inf, np.nan), 0).reset_index(drop=True)
X_test = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\UNSW-NB15\Xval.csv")
X_test = X_test.replace((np.inf, -np.inf, np.nan), 0).reset_index(drop=True)
y_train = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\UNSW-NB15\y.csv")
y_test = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\UNSW-NB15\yval.csv")

clf = []
clf = svm.SVC(kernel='rbf')
clf.fit(X_train, y_train)
predicted_label = clf.predict(X_test)

Precision_Score = precision_score(y_test, predicted_label,  average="macro")
Recall_Score = recall_score(y_test, predicted_label,  average="macro")
F1_Score = f1_score(y_test, predicted_label,  average="macro")
MCC_Score = matthews_corrcoef(y_test, predicted_label)
print(Precision_Score.mean()*100)
print(Recall_Score.mean()*100)
print(F1_Score.mean()*100)
print(MCC_Score*100)
```

```
C:\Users\Mbona\Anaconda3\lib\site-packages\sklearn\utils\validation.py:578: DataConversionWarning: A column-vector y was passed
when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
95.55758059004442
80.5
85.56372623422124
74.55216181966435
```

Figure 8.1: OCSVM ML for discovering zero-day network intrusion attacks results

The results in Figure 8.1 and in Appendix E.1.1 demonstrate that, overall, SSML algorithms produced better results when using significant features identified by the BL method, that is, Experiment 1. The OCSVM SSML achieved the best overall results with F_1 and MCC scores of 86% and 75% respectively. Therefore, OCSVM was able to discover zero-day network intrusion attacks based on the features identified by BL method.

Implementation results using the CICDDOS2019, IoT Intrusion-2020, and CIRACICDOHBRW2020 data sets are presented in Appendices E.1.2, E.1.3, and E.1.4 respectively.

8.3.1.6 Results discussion - NIDSs prototype

This section discusses the experimental results in discovering zero-day network intrusion attacks following the CySecML methodology. In the experiments conducted, four network intrusion data sets discussed in this thesis were used, whereby each SSML algorithm was evaluated using features identified by Benford's law (i.e., Experiment 1) versus features identified by other authors (i.e., Experiment 2). Throughout the experiments, features identified by Benford's law method produced better performance results than those identified by the respective authors.

The differences in the results can be attributed to differences in significant features used as input into the SSML algorithm. Specifically, Benford's law deemed features such as destination and source IP addresses to be redundant, whereas the respective authors deemed such features as significant.

Features such as IP addresses may be useful for signature-based detection methods, whereby incoming network traffic is compared with a database of known malicious network traffic. In the case of a zero-day network intrusion attack with an unknown signature, it is difficult, if not impossible, to use features such as IP addresses to differentiate between benign and unknown zero-day attack. Thus, using such a feature into an ML-based cybersecurity solution can negatively impact its performance in discovering complex attacks such as zero-day attacks. This thesis found features that describe network flow duration, flow inter-arrival times, packets, segment size, TCP base sequence number and the number of connections of the same source or destination address to be significant features for the intelligent discovery of zero-day network intrusion attacks. The findings of this thesis in relation to significant features that should be monitored to intelligently discover zero-day attacks are consistent with those by Chiba et al. [160], Zavrak et al. [175] and Zoppi et al. [179] among others. However, this thesis used a simple Benford's law method as a feature selection method. From a cybersecurity perspective, features such as packets and network traffic flow provide more insight into network traffic behaviour; thus, they can be used to identify anomalies.

Based on the results in Tables E.1, E.1.2, E.1.3, and E.1.4 the OCSVM model produced best results in terms of F_1 and MCC scores followed by label spreading and propagation. The Gaussian mixture model was generally the least-performing model for discovering zero-day network intrusion attacks. By benchmarking the results of this thesis against the existing approaches in Table 3.1, it was observed that Duong et al. [168] and Zhu et al. [170] achieved a recall of 89% and an accuracy of 80% respectively for discovering zero-day intrusion attacks using SSML. In this thesis, the OCSVM achieved a precision of 96%, recall of 81%, F_1 score = 86%, and MCC score = 75%, all of which were slightly better. Moreover, the model adopted in this thesis is also comparable to deep learning models such as those of Abdalgawad et al. [180], Hindy et al. [176], and Zavrak et al. [175], which achieved F_1 score of 85%, an accuracy of 99%, and an AUC of 76% respectively.

This section concludes the experiments conducted to intelligently discover zero-day network

intrusion attacks in NIDSs. The next section focuses on the intelligent discovery of malicious bots in OSNs.

8.3.2 Discovering malicious bots in OSNs

8.3.2.1 Experimental setup

This section describes the experiments that the author of this thesis conducted to intelligently discover malicious bots in OSNs. The experimental approach adopted in this section is similar to that used in the previous section for NIDS data sets. The significant features identified by Benford's law differed from those used in the literature to differentiate between human and malicious bots in OSNs. Therefore, the author of this thesis conducted experiments to evaluate the impact of the features used as inputs in a machine learning algorithm. The experiments were performed as follows.

- Experiment 3 uses significant features identified by Benford's law as input into an SSML algorithm.
- Experiment 4 uses all features identified in Table 5.6 as input into an SSML algorithm.

This section provides implementation details of the prototype for discovering malicious bots on OSNs. In this setup, there were 453 labelled data points consisting of 80 benign bots, and 373 malicious bots launching social engineering attacks. In addition, 18 938 unlabelled bot samples were included. The S3VM SSML was used to evaluate the InternetBotDetector component using features identified by Benford's law (Experiment 3), versus past features for classifying human and malicious bot 𝕏 accounts (Experiment 4).

8.3.2.2 Block I - data extraction experiments

The OSN cybersecurity data set was downloaded from the Internet by the author of this thesis, see Table 5.5. The data set by Oentaryo et al. [4] contained labelled benign and malicious bots records with 72 features. The number of data records in this data set was 453, which is less than ten times the number of features. Therefore, additional samples are required to train a ML

algorithm. Additional samples were obtained from Table 5.5 and added to the Oentaryo et al. [4] data set such that the sample size was sufficiently large.

8.3.2.3 Block II - data storage and quality checks experiments

The data sets in Table 5.5 were downloaded as CSV files (in column vectors) and stored in a local drive. In addition, these data sets were examined and found to be of quality, as shown in Section 6.3.5; thus, data cleaning and feature selection were performed next.

8.3.2.4 Block III - data cleaning and feature selection experiments

No data cleaning was required on the OSN data sets in this thesis. Benford's law data set prerequisites was performed as follows;

- *Leading digits from 1 to 9 must all be possible in a data set* : Using the "ROUND()" function in Excel on the data sets in Table 5.5, all digits from 1 to 9 were observable.
- *A data set must occur naturally* : The data sets in Table 5.5 were extracted from $\mathbb{X}$, and contains natural numbers.
- *Numbers should not be sequential* : Numbers were observed by the author of this thesis, and they did not occur in sequential order, see Figures 5.11, 5.12, 5.13 and 5.14.
- *Numbers should not have pre-defined minimum and maximum values [282, 13]* : The data sets in Table 5.5 do not contain pre-defined min or max values, except for the Boolean type of features such as "URL_count".
- *A data set must be large enough (usually one thousand sample size) [219, 182]* : The data sets in Table 5.5 are sufficiently large.

The OSN data set in Table 5.5 satisfied Benford's law prerequisites; hence, Benford's law was selected as a feature selection method. The significant features identified by Benford's law are shown in the below Table 8.2.

Significant feature
Favourites_count
Lists_count
Statuses_count
Status.retweet_count
Friends_count
Followers_count

Table 8.2: BL OSN significant feature selection results

The identified significant features are then used as inputs into a ML algorithm.

8.3.2.5 Block IV - machine learning and decision-making experiments

For illustrative purposes, the S3VM ML algorithm was used to evaluate the InternetBotDetector model. The below figure depicts the Python environment of the S3VM ML algorithm.

```
import numpy as np
import pandas as pd
from sklearn import metrics
from sklearn import semi_supervised
from sklearn import svm
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, f1_score, precision_score, recall_score, classification_report, confusion_matrix, mat

# Load labeled and unlabeled data sets
X_train = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\X.csv")
X_train = X_train.replace((np.inf, -np.inf, np.nan), 0).reset_index(drop=True)
X_test = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\Xval.csv")
X_test = X_test.replace((np.inf, -np.inf, np.nan), 0).reset_index(drop=True)
y_train = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\y.csv")
y_test = pd.read_csv(r"C:\Users\Mbona\Desktop\ScikitLearn Semi-supervised\yval.csv")

# Fit the S3VM classifier using both labeled and unlabeled datah
s3vm_clf = svm.SVC(kernel='linear', probability=True,C=1).fit(X_train, y_train)
predicted_label = s3vm_clf.predict(X_test)

#nEvaluate the model
Precision_Score = precision_score(np.ravel(y_test), predicted_label,  average="macro")
Recall_Score = recall_score(np.ravel(y_test), predicted_label,  average="macro")
F1_Score = f1_score(np.ravel(y_test), predicted_label,  average="macro")
MCC_Score = matthews_corrcoef(np.ravel(y_test), predicted_label)
print(Precision_Score.mean()*100)
print(Recall_Score.mean()*100)
print(F1_Score.mean()*100)
print(MCC_Score*100)
```

```
C:\Users\Mbona\Anaconda3\lib\site-packages\sklearn\utils\validation.py:578: DataConversionWarning: A column-vector y was passed
when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)

73.5231193926846
70.41657626383163
71.75826601333158
60.87803404443719
```

Figure 8.2: S3VM ML for discovering malicious bots on OSN results

The results in Figure 8.2 and in Appendix E.2 demonstrate that, overall, SSML algorithm produced better results when using significant features identified by the BL method, that is, Experiment 3. The S3VM SSML achieved the best overall results with F_1 and MCC scores of 72% and 61% respectively. Both F_1 and MCC scores are below the 80% threshold, therefore, benign and malicious bots could not be distinctively be differentiated.

8.3.2.6 Results discussion - OSNs prototype

This section discusses the experimental results in discovering social engineering attacks following the CySecML methodology. Each SSML algorithm was evaluated using significant features identified by Benford's law (i.e., Experiment 3) versus all features in Table 5.6 (i.e., Experiment 4). The differences in the results between Experiment 3 and Experiment 4 can be attributed to features used as input into the SSML algorithm. In this regard, Benford's law deemed features such as screen_name length and account age to be redundant. This is intuitive as these features do not provide much insights regarding the behaviour of benign and malicious bots. The findings of this thesis in relation to significant features for the intelligent discovery of malicious bots are consistent with those by De Nicola et al. [74], Rodríguez-Ruiz [215] and Nevado-Catalán et al. [332] among others. In addition, benign and malicious bots share common behavioural characteristics such as posting tweets as shown in Table 3.2, thus classifying them can be challenging.

The S3VM model achieved the best results, with a precision rate of 74%, recall of 70%, F_1 score = 72%, and MCC score = 61%, followed by LP and LS. Note that the F_1 and MCC scores are below the 80% threshold, indicating a challenge in distinguishing between benign and malicious bots. By comparing the results with the existing approaches in Tables 3.3 and 3.4, Oentaryo et al. [4] model achieved F_1 score = 74% in discovering social engineering attacks launched by bots using a supervised machine learning approach.

An interesting observation from a cybersecurity perspective is that features that are useful in classifying human and malicious bots in OSNs are not equally useful in classifying benign and malicious bots. Although the prototypes implemented this thesis demonstrate promising results, some limitations of the InternetBotDetector model are noted below.

8.3.3 Limitations of the InternetBotDetector model

The limitations identified by this author associated with implementing the InternetBotDetector model include the following:

- Manual labelling of bot samples: The InternetBotDetector model adopts SSML algorithms that require the labelling of small samples of benign and malicious bot activities. The labelling of bot samples is dependent on the Internet Application Platform (IAP) used in this thesis. In addition, the InternetBotDetector model only solves binary problems - in this case, it determines whether a particular bot activity appears benign or malicious.
- Configuration: In order for the InternetBotDetector model to be integrated into an IAP, it requires configuration by an IT person so that it can extract data from that IAP to intelligently discover malicious bot activities. For example, for the InternetBotDetector model to be the most effective, it requires information about how an IAP normally operates and what type of activity should trigger an alert to the system controller.
- Maintenance: Just like any other ML-based cybersecurity solution, the InternetBotDetector model requires manual updating in terms of using the most recent bot data sets and configuration. Therefore, maintenance by IT staff is required for the InternetBotDetector model to perform effectively.

8.4 Chapter summary

This chapter presents the prototype implementation results based on the CySecML methodology that was used to intelligently discover zero-day network intrusion attacks in NIDSs and social engineering attacks in OSNs. The OCSVM algorithm produced the best results for discovering zero-day network intrusion attacks based on features identified using Benford's law method. Some significant features identified by Benford's law include network packets and flow features. The S3VM algorithm produced the best results for discovering social engineering attacks, based on the features identified using Benford's law method. Moreover, it was demonstrated that features such as URL_count, which has been shown to be useful for classifying human and malicious 𝕏 accounts, were not equally useful for classifying benign and malicious 𝕏 accounts.

It is important to note that the author of this thesis do not claim that Benford's law is the only feature selection method capable of identifying significant features for assisting ML-based cybersecurity solutions. However, this thesis demonstrated that Benford's law is a viable feature selection method for identifying features indicative of anomalous behaviour in cybersecurity data sets. Although the InternetBotDetector model produced promising results in discovering cyberattacks launched by bots, one of its main limitations is that it requires configuration by an IT person for its integration into an existing Internet Application Platform. Overall, the CySecML methodology was shown to produce promising results in providing guidelines for the development of ML-based cybersecurity solutions. The significance of including data quality checks in the CySecML methodology is to ensure that one can rely on findings based on using a particular data set. The next chapter concludes this thesis.

Conclusion

Chapter 9

Conclusion

9.1 Introduction

This thesis addressed the problem of intelligent discovery of malicious bots on Internet Application Platforms (IAPs), based on anomalous behaviour detection. The problem was addressed by proposing the CySecML methodology, which provides organisations with a framework of developing ML-based cybersecurity solutions. Proof-of-concept for this methodology was implemented in Chapter 8. Two different IAPs - online social networks (OSNs) and network intrusion detection systems (NIDSs) - were chosen to investigate bots cyberattacks. In this concluding chapter, the author of this thesis revisit the main research problem and the research questions examined in this thesis to determine the extent to which they have been addressed. Finally, the main contributions and limitations of the thesis, as well as suggestions for future research are presented. Next, the problem statement is revisited.

9.2 Revisiting the problem statement

The main focus of this thesis was to address the problem of intelligent discovery of cyberattacks launched by malicious bots on IAPs, based on anomalous behaviour detection. The cyberattacks considered in this thesis are social engineering attacks in OSNs, and zero-day network intrusion attacks in NIDSs.

The research problem was addressed by answering the following research questions:

- **Research Question 1: How to develop or enhance state-of-the-art machine learning based cybersecurity solutions to countermeasure cyberattacks launched by bots?**

  The CySecML methodology developed in Chapter 4 aims to provide organisations with a framework of developing ML-based cybersecurity solutions. The CySecML methodology enhances existing cybersecurity solutions by implementing data quality checks on cybersecurity data sets, and using Benford's law (BL) as a feature selection method on cybersecurity data sets. Chapter 8 was dedicated to the implementation of CySecML methodology prototypes to intelligently discover cyberattacks launched by bots such as social engineering and zero-day network intrusion attacks. In this process, the author of this thesis implemented various semi-supervised machine learning (SSML) algorithms.

  The SSML algorithms were trained using labelled and unlabelled data, based on the significant features identified by the Benford's law method. In the case of discovering cyberattacks such as social engineering attacks in OSNs, the S3VM SSML algorithm achieved the best results with a recall of 70% and F_1 score of 72% based on the features identified by Benford's law in Table E.5. Using all the features in Table 7.6, which have been shown to be useful in differentiating between human and malicious bots, did not improve the model's performance in classifying benign and malicious bots. In this case, recall was 68% and F_1 score of 67%. This suggests that significant features for classifying humans and malicious bots may not be equally effective for classifying benign and malicious bots [294]. The results in Table E.5 demonstrate that SSML approaches are effective when features indicative of anomalous behaviour are carefully chosen, and they are comparable to the existing approaches in Tables 3.3 and 3.4. Although the results were not conclusive for differentiating benign and malicious bots based on the threashold of 80%. However, most importantly, SSML algorithms overcome the challenge of obtaining large amounts of labelled attack cybersecurity data to train ML-based cybersecurity solutions.

  For the discovery of cyberattacks such as zero-day network intrusion attacks in NIDSs, the OCSVM SSML algorithm achieved the best results with a recall of 81%, MCC score of 75%, and F_1 score of 86%, based on features identified by Benford's law. The results in Table E.1 demonstrate that SSML approaches are effective when benchmarked against

the existing methods in Table 3.1, such as deep learning approaches.

- **Research Question 2: What type of features found on Internet Application Platforms that should be monitored to effectively discover cyberattacks launched by bots?**

The three broad categories of feature types found in OSNs and NIDSs are either *static*, *semi-static* or *dynamic* in nature (see Chapter 7). For example, features that describe user details such as user ID number are considered static, as they do not change from time to time [333, 334]. Features that describe user activities, such as friends_count in OSNs, are considered dynamic, as they change from time to time [333, 334]. An example of static features found in OSNs is screen_name length, which describes the number of characters in an account profile. This feature does not change its value from time to time, unless the account owner decides to change the screen_name. Similarly, in NIDSs, features such as source and destination Internet Protocol (IP) addresses that describe device information are static. However, features that describe user activities such as status_count (which counts the number of posts made by an OSN account), are dynamic and their values change over time. Likewise, NIDSs features such as network traffic flow duration, which describes the duration of data flow, are dynamic in nature.

Features indicative of malicious bot activities in OSNs and NIDSs were identified using Benford's law of anomalous numbers. In the literature, Benford's law has been shown to successfully detect cybercrimes such as denial-of-service (DoS) attacks in NIDSs [187] and spamming in OSNs [219]. One of the contributions made by this thesis was to demonstrate that Benford's law is an effective and scalable feature selection method for cybersecurity data sets because it is able to efficiently deal with big data volumes that evolve continuously as in NIDSs and OSNs.

Chapter 7 investigated features indicative of bots anomalous behaviour for discovering social engineering and zero-day network intrusion attacks. For the OSN case, it was found that dynamic features such as status_count, friends_count, followers_count, status.retweet_count, favourites_count, and lists_count were significant for discovering social engineering attacks. Although features such as URL_count, status.reply_count, hashtag_count, status.favourite_count, and account age (in days) were also dynamic, benign and malicious bot behaviours displayed indistinguishable behaviour on these fea-

tures. This is because, although these features are dynamic, they take particular values, for example, binary types of features, which makes it impossible to distinctively differentiate between benign and malicious bot activities. Similarly, static features such as screen_name length could not distinguish between benign and malicious bot behaviours. Although the author of this thesis used a 𝕏 data set to conduct the experiments, the findings of this thesis are also applicable to other OSNs platforms such as Facebook, given that raw meta data was used.

For the NIDSs case, the author of this thesis found that dynamic features, including flow duration, flow inter-arrival times, packets, and segment sizes are significant for discovering zero-day network intrusion attacks. Static or semi-static features such as the Internet Protocol were found to be redundant for discovering zero-day network intrusion attacks. It is the opinion of this thesis author that, it is difficult to know beforehand which Internet Protocols cybercriminals will utilise to launch a zero-day network intrusion attack. A similar finding was reported by [178, 160].

- **Research Question 3: What cyberattacks are launched by bots currently found on the Internet Application Platforms?**

As demonstrated in Chapter 2, numerous cyberattacks are launched by different types of bots on different IAPs. These cyberattacks include misinformation or fake news, spam, DoS or DDoS, deception, malware, ransomware, phishing, social engineering, trolling, sybil, and zero-day attacks. These cyberattacks were identified through a systematic literature review (SLR) process based on the preferred reporting items for systematic reviews and meta-analyses (PRISMA) framework. To conduct this SLR, the author of this thesis focused on relevant articles published between January 2010 and December 2020, see [85] for further details.

cyberattacks that include misinformation or fake news, social engineering, and spamming were found to be prominent cyberattacks launched by bots in OSNs. Likewise, cyberattacks such as zero-day network intrusion attacks were also deemed of significant importance as they are globally increasing fast [188, 335]. Distributing misinformation or fake news [106, 50] on OSNs involves the spread of falsified information with the intention of swaying public opinion, whereas spamming is the act of sending unsolicited information

[336, 50]. Social engineering attacks are manipulation methods performed through fake social engagements to trick humans into committing security mistakes [142, 326]. Given the rapidly growing number of users on OSNs, humans are vulnerable to cybercrimes launched by bots, see studies such as [51, 106]. Thus, it is of utmost importance that humans are protected from the cybercrimes launched by bots. It is crucial that CTI procedure that provide organisations with strategic solutions to combat cyber threats [40, 77] be continuously enhanced to enable intelligent discovery of new unknown (zero-day) network intrusion attacks. Zero-day network intrusion attacks can be launched in various ways [71, 59] and they hence were deemed a crucial security problem to be addressed in this thesis.

- **Research Question 4: What are the important data quality characteristics to be considered on cybersecurity data sets when designing machine learning based cybersecurity solutions?**

  Chapter 6 focused on examining the data quality characteristics of cybersecurity data sets. Ensuring data quality is crucial when designing ML-based cybersecurity solutions so that the results can be reliable. Various data quality characteristics are found in the literature across different fields of study, but not specifically for ML-based cybersecurity problems. The contribution of this thesis in this regard was proposing data quality characteristics, including availability, usability, reliability, relevance, data size, and presentation quality for cybersecurity data sets. The NIDS and OSN cybersecurity data sets used in this thesis were examined in accordance with these data quality characteristics, and they were found to be of quality by satisfying all these characteristics. Ideally, a cybersecurity data set must satisfy all these characteristics to be deemed of quality. The main challenge in implementing these data quality characteristics is the difficulty to quantify them, except for data size characteristic.

9.3 Summary of main contributions

The main contribution of the current thesis was its proposition of the CySecML methodology for the intelligent discovery of cyberattacks launched by bots. The author of this thesis believe

that the CySecML methodology is unique from existing methodologies given its unique features on extracting cybersecurity data sets, checking data quality on cybersecurity data sets and using Benford's law as a feature selection method on cybersecurity data sets. The CySecML methodology provides a framework for enhancing cybersecurity solutions based on machine learning (ML). ML-based solutions are generally considered intelligent, as they utilise predictive or classification algorithms to make informed decisions [236, 37].

The CySecML methodology consists of data preparation and InternetBotDetector model components. The data preparation component provides guidelines for obtaining or creating cybersecurity data for training ML-based cybersecurity solutions. In addition, this component provides characteristics of a data set that should be examined to ensure that the data used to train a model is of data quality. Finally, a unique feature selection method based on Benford's law was proposed to identify significant features that are indicative of anomalous behaviours. Internet Application Platforms, such as OSNs and NIDSs, face high-dimensional imbalanced data problems that can cause the process of identifying significant features to be computationally expensive. The output or results of the data preparation component are subsequently used as inputs for the InternetBotDetector model component.

The InternetBotDetector model component is based on ML algorithms and aims to intelligently discover cyberattacks launched by bots. The model was implemented to discover bot cyberattacks such as social engineering and zero-day network intrusion attacks. The main original contributions of this thesis can be summarised as follows:

- It proposes a bot taxonomy from a cybersecurity perspective that is based on the bot type, domain type, bot cybercrime, and mitigation or discovery methods [85]. The current literature on bots lacks a holistic overview of the different types of bots and the cybercrimes they commit. The proposed bot taxonomy is based on a systematic literature review process that defines the search strategy, inclusion and exclusion criteria, eligibility, and quality assessment. Herein, various research gaps such as the classification of benign and malicious bots were identified as not all bots found in IAPs are malicious [294]. Although the prototypes conducted in this thesis used OSN and NIDS platforms, malicious bots are also present in other IAPs such as websites that can benefit from the CySecML methodology.

- This thesis demonstrates that Benford's law is an effective and efficient feature selection method on cybersecurity data sets. According to the author of this thesis, Benford's law was applied as a feature selection method on cybersecurity data sets for the first time in this thesis. The prototypes conducted in this thesis demonstrate that Benford's law can improve the performance of cybersecurity solutions such as the InternetBotDetector model in discovering bot cyberattacks. The significant features identified by Benford's law on OSNs for discovering social engineering attacks were demonstrated to improve the performance of the proposed model. Similarly, the significant features identified by Benford's law on NIDSs for the discovery of zero-day network intrusion attacks also improved the performance of the InternetBotDetector model. Benford's law as a feature selection method was benchmarked against various other feature selection methods, including ensemble random forest, and the results were consistent [189, 295]. Although the author of this thesis does not claim that Benford's law outperforms existing feature selection methods, this thesis suggests that it is a straightforward method to implement that can be expected to reduce the computational cost of identifying significant features on cybersecurity data sets, that are generally considered as big data and imbalanced.

- This thesis offered a pseudo code representation of the CySecML methodology that provides simplified steps of implementing this methodology. The pseudo code representation provides guidelines of how this methodology can be implemented in practice. The data preparation component aims to overcome the cybersecurity challenge of obtaining real-world attack data for the purposes of training ML-based cybersecurity solutions. This challenge is mainly attributed to privacy laws and regulations pertaining to sensitive data.

This thesis has made a significant contribution towards addressing the problem of intelligent discovery cyberattacks launched by malicious bots on IAP based on anomalous behaviour detection. The next section highlights noteworthy limitations of this thesis.

9.4 Limitations of the study

In this section, the limitations of this thesis are summarised as a retrospective exercise.

- This thesis limited its focus to cyberattacks launched by bots - and not humans - on Internet Application Platforms (IAPs). In addition, the study used NIDS and OSN data

sets with their original features. It did not create new features, although one can create a new feature such as "flow_bytes to packets", which is a ratio of network flow_bytes and flow packets.

- In as much as using existing data sets for the purposes of training a machine learning algorithm is generally a good practice for benchmarking purposes, some of the data sets used in this thesis such as the UNSW-NB15 and the Oentaryo et al. [4] can be considered out of date (created in 2015 and obtained in 2016 respectively).

- The proposed CySecML methodology was applied to only two different IAPs - online social networks and network intrusion detection systems - to conduct experiments for discovering bot social engineering and zero-day network intrusion attacks respectively. Thus, the results obtained may be biased towards these IAPs as opposed to other IAPs.

Considering the above limitations and the findings of this thesis, it is evident that there is room for future research.

9.5 Future work

Although the proposed CySecML methodology addressed the main research problem and questions, there are opportunities to extend this thesis for future research.

- Further investigation can be conducted on the performance abilities of the InternetBot-Detector model on different IAPs, as well as on other features found on IAPs that can improve the discovery of cyberattacks launched by bots, in particular for differentiating between benign and malicious bots. For example, websites and other online social network platforms such as Facebook and Instagram can be considered.

- An attempt must be made to overcome the challenge of obtaining real-world attack data for the purposes of training a cybersecurity solution based on machine learning. There is an opportunity to leverage on the significant features identified by this thesis to generate synthetic data that is representative of cyberattacks launched by bots. Using the IoT Intrusion-2020 cybersecurity data set as an example, features such as flow duration and packets were demonstrated to differentiate between benign and botnet attacks. Therefore,

such features can be used to generate synthetic data with the same statistical properties as the original data.

- Further research is required to standardise the process of quantifying the data quality characteristics presented in this study. For instance, develop a data quality scorecard model for quantifying data quality of a data set. Cybersecurity data sets such as those from OSN and NIDS are generally considered big data; therefore, the ability to quantify these data quality characteristics efficiently will be crucial for big data platforms.

- Recent developments in generative artificial intelligence (GenAI) techniques such as generative adversarial networks (GANs) create new challenges such as "deepfake" for the cybersecurity community. GenAI techniques can generate synthetic data in the form of text, audio, images, videos and so on to appear as original data, at least to the human eye. This imposes a new challenge on the cybersecurity community to differentiate between bot-generated and human-generated content, for example, between content generated by AI tools such as ChatGPT and content generated by humans. Bot-generated content may increase the incidence of cybercrimes such as spamming, trolling, phishing, and identity deception (i.e., deepfake). Therefore, cybersecurity solutions will need to be refined further to mitigate the ever-growing gravity of cybercrimes. The convergence of AI and cybersecurity will be the next fascinating area of research in the coming years.

9.6 Concluding remarks

The research problem as defined in this thesis posed a unique and interesting challenge because bots continuously produce cybersecurity threats that cannot be ignored. Collaborating with my colleagues at the University of Pretoria, working on the SASUF project between Sweden and South Africa, and attending the BRICS summit in Brazil proved to be a fruitful experience. Finally, it is hoped that this thesis will pave the way for future research to gain an in-depth understanding of both benign and malicious bots from a cybersecurity perspective.

Bibliography

[1] H. M. Harb, A. A. Zaghrot, M. A. Gomaa, and A. S. Desuky, "Selecting optimal subset of features for intrusion detection systems," *Advances in Computational Sciences and Technology (ACST)*, 2011. [Online]. Available: https://www.ripublication.com/Volume/acstv4n2.htm

[2] N. Moustafa and J. Slay, "The Significant Features of the UNSW-NB15 and the KDD99 Data Sets for Network Intrusion Detection Systems," in *2015 4th International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS)*. IEEE, Nov 2015. [Online]. Available: https://doi.org/10.1109%2Fbadgers.2015.014

[3] N. F. L. M. Rosely, A. M. Zain, and Y. Yusoff, "Feature Selection of High Dimensional Data Using Hybrid FSA-IG," *IOP Conference Series: Materials Science and Engineering*, vol. 864, no. 1, p. 012066, May 2020. [Online]. Available: https://doi.org/10.1088%2F1757-899x%2F864%2F1%2F012066

[4] R. J. Oentaryo, A. Murdopo, P. K. Prasetyo, and E.-P. Lim, "On Profiling Bots in Social Media," in *Lecture Notes in Computer Science*. Springer International Publishing, 2016, pp. 92–109. [Online]. Available: https://doi.org/10.1007%2F978-3-319-47880-7_6

[5] K.-C. Yang, O. Varol, P.-M. Hui, and F. Menczer, "Scalable and Generalizable Social Bot Detection through Data Selection," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, p. 10961103, Apr. 2020. [Online]. Available: http://dx.doi.org/10.1609/aaai.v34i01.5460

[6] M. Mazza, S. Cresci, M. Avvenuti, W. Quattrociocchi, and M. Tesconi, "RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter," in *Proceedings of the 10th ACM Conference on Web Science*. ACM, Jun 2019. [Online]. Available: https://doi.org/10.1145%2F3292522.3326015

[7] K.-C. Yang, O. Varol, C. A. Davis, E. Ferrara, A. Flammini, and F. Menczer, "Arming the public with artificial intelligence to counter social bots," *Human Behavior and Emerging Technologies*, vol. 1, no. 1, pp. 48–61, Jan 2019. [Online]. Available: https://doi.org/10.1002%2Fhbe2.115

[8] I. Ullah and Q. H. Mahmoud, "A Technique for Generating a Botnet Dataset for Anomalous Activity Detection in IoT Networks," in *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, Oct 2020. [Online]. Available: https://doi.org/10.1109%2Fsmc42975.2020.9283220

[9] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *2019 International Carnahan Conference on Security Technology (ICCST)*. IEEE, Oct 2019. [Online]. Available: https://doi.org/10.1109%2Fccst.2019.8888419

[10] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. H. Lashkari, "Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic," in *2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*. IEEE, Aug 2020. [Online]. Available: https://doi.org/10.1109%2Fdasc-picom-cbdcom-cyberscitech49142.2020.00026

[11] L. Cai and Y. Zhu, "The challenges of data quality and data quality assessment in the big data era," in *Data Science Journal*, vol. 14. Committee on Data for Science and Technology, 2015. [Online]. Available: https://doi.org/10.5334/dsj-2015-002

[12] F. Ridzuan and W. M. N. W. Zainon, "A Review on Data Quality Dimensions for Big Data," *Procedia Computer Science*, vol. 234, p. 341348, 2024. [Online]. Available: http://dx.doi.org/10.1016/j.procs.2024.03.008

[13] S. Miller, *Benford's Law*. Princeton University Press, 2015.

[14] V. Rajaraman, "A Concise History of the Internet-I," *Resonance*, vol. 27, no. 11, pp. 1841–1856, 2022. [Online]. Available: https://doi.org/10.1007/s12045-022-1483-2

[15] L. Kleinrock, "An early history of the internet [History of Communications]," *IEEE Communications Magazine*, vol. 48, no. 8, pp. 26–36, Aug 2010. [Online]. Available: https://doi.org/10.1109%2Fmcom.2010.5534584

[16] J. Kremling and A. M. S. Parker, *Cyberspace, cybersecurity, and cybercrime*. Sage Publications, 2017.

[17] A. S. K. Pathan, *Securing Social Networks in Cyberspace*. CRC Press, 2021.

[18] J. M. Hatfield, "Social engineering in cybersecurity: The evolution of a concept," *Computers & Security*, vol. 73, pp. 102–113, Mar 2018. [Online]. Available: https://doi.org/10.1016%2Fj.cose.2017.10.008

[19] M. H. Miraz, M. Ali, P. S. Excell, and R. Picking, "A review on Internet of Things (IoT), Internet of Everything (IoE) and Internet of Nano Things (IoNT)," in *2015 Internet Technologies and Applications (ITA)*. IEEE, Sep 2015. [Online]. Available: https://doi.org/10.1109%2Fitecha.2015.7317398

[20] W. Mardini, Y. Khamayseh, and A. Smadi, "Messenger Bot for IoT devices," in *Proceedings of the 9th International Conference on Information Management and Engineering*. ACM, Oct 2017. [Online]. Available: https://doi.org/10.1145%2F3149572.3149611

[21] A. Shevat, *Designing bots: Creating conversational experiences*. O'Reilly Media, Inc, 2017.

[22] N. Chavoshi and A. Mueen, "Model Bots, not Humans on Social Media," in *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, Aug 2018. [Online]. Available: https://doi.org/10.1109%2Fasonam.2018.8508279

[23] M. Chatterjee, A. S. Namin, and P. Datta, "Evidence Fusion for Malicious Bot Detection in IoT," in *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec 2018. [Online]. Available: https://doi.org/10.1109%2Fbigdata.2018.8621895

[24] B. Stephens, A. Shaghaghi, R. Doss, and S. S. Kanhere, "Detecting Internet of Things Bots: A Comparative Study," *IEEE Access*, vol. 9, pp. 160 391–160 401, 2021. [Online]. Available: https://doi.org/10.1109%2Faccess.2021.3130714

[25] R. Kar and R. Haldar, "Applying Chatbots to the Internet of Things: Opportunities and Architectural Elements," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 11, 2016. [Online]. Available: https://doi.org/10.14569%2Fijacsa.2016.071119

[26] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *26th USENIX security symposium (USENIX Security 17)*, 2017, pp. 1093–1110. [Online]. Available: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/antonakakis

[27] C. Frank, S. Jarocki, C. Nance, and W. Pauli, "Protecting IoT Devices from the Mirai Botnet," *Journal of Information Systems Applied Research*, vol. 11, no. 2, p. 11, 2018. [Online]. Available: http://jisar.org/2018-11/n2/JISARv11n2.pdf#page=33

[28] C. Lebeuf, A. Zagalsky, M. Foucault, and M.-A. Storey, "Defining and Classifying Software Bots: A Faceted Taxonomy," in *2019 IEEE/ACM 1st International Workshop on Bots in Software Engineering (BotSE)*. IEEE, May 2019. [Online]. Available: https://doi.org/10.1109%2Fbotse.2019.00008

[29] L. Erlenhov, F. G. D. O. Neto, and P. Leitner, "An empirical study of bots in software development: Characteristics and challenges from a practitioner's perspective," in *Proceedings of the 28th ACM Joint Meeting European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. Association for Computing Machinery, Inc, Nov 2020, pp. 445–455. [Online]. Available: https://doi.org/10.1145/3368089.3409680

[30] S. Stieglitz, F. Brachten, B. Ross, and A. K. Jung, "Do social bots dream of electric sheep? a categorisation of social media bot accounts," *Proceedings of the 28th Australasian Conference on Information Systems, ACIS 2017*, pp. 1–11, 2017. [Online]. Available: https://doi.org/10.48550/arXiv.1710.04044

[31] E. Alothali, N. Zaki, E. A. Mohamed, and H. Alashwal, "Detecting Social Bots on Twitter: A Literature Review," *Proceedings of the 2018 13th International Conference on Innovations in Information Technology, IIT 2018*, pp. 175–180, 2019. [Online]. Available: https://doi.org/10.1109/INNOVATIONS.2018.8605995

[32] S. Sivakorn, J. Polakis, and A. D. Keromytis, "I'm not a human: Breaking the Google reCAPTCHA," *Black Hat*, vol. 14, pp. 1–12, 2016.

[33] S. Rozga, *Practical Bot Development - Designing and Building Bots with Node.js and Microsoft Bot Framework*. Apress, 2018.

[34] S. Barbon, G. F. Campos, G. M. Tavares, R. A. Igawa, M. L. Proença, and R. C. Guido, "Detection of human, legitimate bot, and malicious bot in online social networks based on wavelets," *ACM Transactions on Multimedia Computing, Communications and Applications*, vol. 14, no. 1s, 2018. [Online]. Available: https://doi.org/10.1145/3183506

[35] S. Dua and X. Du, *Data mining and machine learning in cybersecurity*. CRC press, 2016.

[36] Y. Diogenes, *Cybersecurity: Attack and Defense Strategies*. Packt Publishing, 2018.

[37] I. H. Sarker, A. S. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big Data*, vol. 7, no. 1, Dec 2020. [Online]. Available: https://doi.org/10.1186/s40537-020-00318-5

[38] A. Moustafa, *Cybersecurity and Cognitive Science*. Elsevier, 2022. [Online]. Available: https://doi.org/10.1016%2Fc2020-0-03849-1

[39] M. S. Abu, S. R. Selamat, A. Ariffin, and R. Yusof, "Cyber threat intelligence Issue and challenges," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 10, pp. 371–379, 2018. [Online]. Available: https://doi.org/10.11591/ijeecs.v10.i1.pp371-379

[40] A. N. Irfan, S. Chuprat, M. N. ri Mahrin, and A. Ariffin, "Taxonomy of Cyber Threat Intelligence Framework," in *International Conference on ICT Convergence*. IEEE Computer Society, 2022, pp. 1295–1300. [Online]. Available: https://doi.org/10.1109/ICTC55196.2022.9952616

[41] D. Schlette, M. Caselli, and G. Pernul, "A Comparative Study on Cyber Threat Intelligence: The Security Incident Response Perspective," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2525–2556, 2021. [Online]. Available: https://doi.org/10.1109%2Fcomst.2021.3117338

[42] M. Sahrom Abu, S. Rahayu Selamat, A. Ariffin, and R. Yusof, "Cyber Threat Intelligence Issue and Challenges," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 10, no. 1, p. 371, Apr. 2018. [Online]. Available: http://dx.doi.org/10.11591/ijeecs.v10.i1.pp371-379

[43] C. Freitas, F. Benevenuto, S. Ghosh, and A. Veloso, "Reverse Engineering Socialbot Infiltration Strategies in Twitter," in *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ACM, Aug 2015. [Online]. Available: https://doi.org/10.1145%2F2808797.2809292

[44] O. Varol, C. A. Davis, F. Menczer, and A. Flammini, "Feature Engineering for Social Bot Detection," *Feature Engineering for Machine Learning and Data Analytics*, pp. 311–334, 2018. [Online]. Available: https://doi.org/10.1201/9781315181080-12

[45] A. J. V. Fonseca, F. J. H. Cabral, G. J. J. Costa, and R. C. Ghedini, "Bot development for social engineering attacks on Twitter," *arXiv*, vol. 1, no. 2013, 2020. [Online]. Available: https://doi.org/10.48550/arXiv.2007.11778

[46] G. Nalinipriya and M. Asswini, "A survey on vulnerable attacks in online social networks," *Proceedings 2015 - IEEE International Conference on Innovation, Information in Computing Technologies, ICIICT 2015*, 2016. [Online]. Available: https://doi.org/10.1109/ICIICT.2015.7396102

[47] U. Can and B. Alatas, "A new direction in social network analysis: Online social network analysis problems and applications," *Physica A: Statistical Mechanics and its Applications*, vol. 535, pp. 122–372, 2019. [Online]. Available: https://doi.org/10.1016/j.physa.2019.122372

[48] C. Chapman, *Traffic performance testing in the network*. Syngress Publishing, 2016.

[49] E. Etuh, F. S. Bakpo, and E. A.H, "Social Media Network Attacks and their Preventive Mechanisms: A Review," *arXiv preprint arXiv:2201.03330*, pp. 59–74, 2021. [Online]. Available: https://doi.org/10.5121/csit.2021.112405

[50] A. Bondielli and F. Marcelloni, "A survey on fake news and rumour detection techniques," *Information Sciences*, vol. 497, pp. 38–55, 2019. [Online]. Available: https://doi.org/10.1016/j.ins.2019.05.035

[51] K. Sharma, F. Qian, H. Jiang, N. Ruchansky, M. Zhang, and Y. Liu, "Combating fake news: A survey on identification and mitigation techniques," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 3, 2019. [Online]. Available: https://doi.org/10.1145/3305260

[52] M. C. Forelle, P. N. Howard, A. Monroy-Hernandez, and S. Savage, "Political Bots and the Manipulation of Public Opinion in Venezuela," *SSRN Electronic Journal*, pp. 1–8, 2015. [Online]. Available: https://doi.org/10.2139/ssrn.2635800

[53] L. M. Adahali and M. Hall, "Application of the Benford's law to Social bots and Information Operations activities," *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment, Cyber SA 2020*, 2020. [Online]. Available: https://doi.org/10.1109/CyberSA49311.2020.9139709

[54] B. Kollanyi, P. N. Howard, and S. C. Woolley, "Bots and automation over Twitter during the first US presidential debate," *COMPROP Data Memo*, vol. 1, pp. 1–4, 2016.

[55] M. Wiesenberg and R. Tench, "Deep strategic mediatization: Organizational leaders knowledge and usage of social bots in an era of disinformation," *International Journal of Information Management*, vol. 51, p. 102042, Apr. 2020. [Online]. Available: http://dx.doi.org/10.1016/j.ijinfomgt.2019.102042

[56] J. Fernquist, L. Kaati, and R. Schroeder, "Political bots and the swedish general election," *2018 IEEE International Conference on Intelligence and Security Informatics, ISI 2018*, pp. 124–129, 2018. [Online]. Available: https://doi.org/10.1109/ISI.2018.8587347

[57] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, 2019. [Online]. Available: https://doi.org/10.1016/j.cose.2019.06.005

[58] S. Romanosky, "Examining the costs and causes of cyber incidents," *Journal of Cybersecurity*, vol. 2, no. 2, pp. 121–135, 2016. [Online]. Available: https://doi.org/10.1093/cybsec/tyw001

[59] U. K. Singh, C. Joshi, and D. Kanellopoulos, "A framework for zero-day vulnerabilities detection and prioritization," *Journal of Information Security and Applications*, vol. 46, pp. 164–172, 2019. [Online]. Available: https://doi.org/10.1016/j.jisa.2019.03.011

[60] X. Wang, K. Sun, A. Batcheller, and S. Jajodia, "Detecting '0-Day' Vulnerability: An Empirical Study of Secret Security Patch in OSS," *Proceedings - 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2019*, pp. 485–492, 2019. [Online]. Available: https://doi.org/10.1109/DSN.2019.00056

[61] R. Azmi, W. Tibben, and K. T. Win, "Review of cybersecurity frameworks: context and shared concepts," *Journal of Cyber Policy*, vol. 3, no. 2, pp. 258–283, 2018. [Online]. Available: https://doi.org/10.1080/23738871.2018.1520271

[62] P. P. Roy, "A High-Level Comparison between the NIST Cyber Security Framework and the ISO 27001 Information Security Standard," *2020 National Conference on Emerging Trends on Sustainable Technology and Engineering Applications, NCETSTEA 2020*, vol. 53, pp. 27 001–27 003, 2020. [Online]. Available: https://doi.org/10.1109/NCETSTEA48365.2020.9119914

[63] L. A. Gordon, M. P. Loeb, and L. Zhou, "Integrating cost-benefit analysis into the NIST cybersecurity framework via the gordon-loeb model," *Journal of Cybersecurity*, vol. 6, no. 1, pp. 1–8, 2020. [Online]. Available: https://doi.org/10.1093/CYBSEC/TYAA005

[64] R. Kwon, T. Ashley, J. Castleberry, P. Mckenzie, and S. N. Gupta Gourisetti, "Cyber Threat Dictionary Using MITRE ATT&CK Matrix and NIST Cybersecurity Framework Mapping," in *2020 Resilience Week (RWS)*. IEEE, Oct. 2020. [Online]. Available: http://dx.doi.org/10.1109/rws50334.2020.9241271

[65] M. Latah, "Detection of malicious social bots: A survey and a refined taxonomy," *Expert Systems with Applications*, vol. 151, p. 113383, 2020. [Online]. Available: https://doi.org/10.1016/j.eswa.2020.113383

[66] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, 2019. [Online]. Available: https://doi.org/10.1186/s42400-019-0038-7

[67] J. L. Leevy and T. M. Khoshgoftaar, "A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data," *Journal of Big Data*, vol. 7, no. 1, 2020. [Online]. Available: https://doi.org/10.1186/s40537-020-00382-x

[68] R. R. Rout, G. Lingam, and D. V. Somayajulu, "Detection of Malicious Social Bots Using Learning Automata with URL Features in Twitter Network," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 4, pp. 1004–1018, 2020. [Online]. Available: https://doi.org/10.1109/TCSS.2020.2992223

[69] M. Zabihimayvan, R. Sadeghi, H. N. Rude, and D. Doran, "A soft computing approach for benign and malicious web robot detection," *Expert Systems with Applications*, vol. 87, pp. 129–140, 2017. [Online]. Available: https://doi.org/10.1016/j.eswa.2017.06.004

[70] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, Feb 2020. [Online]. Available: https://doi.org/10.1016%2Fj.jisa.2019.102419

[71] D. Cuppah, D. Chea Cuppah, and M. Hanumanthappa, "Design and Analysis of a Hybrid Security Framework for Zero-Day Attack," *International Journal of Applied Engineering Research*, vol. 14, no. May, pp. 140–144, 2020.

[72] F. Abri, S. Siami-Namini, M. A. Khanghah, F. M. Soltani, and A. S. Namin, "Can Machine/Deep Learning Classifiers Detect Zero-Day Malware with High Accuracy?" *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, pp. 3252–3259, 2019. [Online]. Available: https://doi.org/10.1109/BigData47090.2019.9006514

[73] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Machine Learning*, vol. 109, no. 2, pp. 373–440, Nov 2019. [Online]. Available: https://doi.org/10.1007%2Fs10994-019-05855-6

[74] R. De Nicola, M. Petrocchi, and M. Pratelli, "On the efficacy of old features for the detection of new bots," *Information Processing and Management*, vol. 58, no. 6, pp. 1–15, 2021. [Online]. Available: https://doi.org/10.1016/j.ipm.2021.102685

[75] D. Ramalingam and V. Chinnaiah, "Fake profile detection techniques in large-scale online social networks: A comprehensive review," *Computers and Electrical Engineering*, vol. 65, no. 3, pp. 165–177, 2018. [Online]. Available: https://doi.org/10.1016/j.compeleceng.2017.05.020

[76] S. Thudumu, P. Branch, J. Jin, and J. J. Singh, "A comprehensive survey of anomaly detection techniques for high dimensional big data," *Journal of Big Data*, vol. 7, no. 1, 2020. [Online]. Available: https://doi.org/10.1186/s40537-020-00320-x

[77] M. Landauer, F. Skopik, M. Wurzenberger, W. Hotwagner, and A. Rauber, "A Framework for Cyber Threat Intelligence Extraction from Raw Log Data," in *IEEE International Conference on Big Data (Big Data)*, 2019, pp. 3200–3209. [Online]. Available: https://doi.org/10.1109/BigData47090.2019.9006328

[78] M. A. Khder, S. Shorman, D. A. Showaiter, A. S. Zowayed, and S. I. Zowayed, "Review Study of the Impact of Artificial Intelligence on Cyber Security," in *2023 International*

Conference on IT Innovation and Knowledge Discovery (ITIKD). IEEE, Mar. 2023. [Online]. Available: http://dx.doi.org/10.1109/itikd56332.2023.10099788

[79] C. Okoli and K. Schabram, "A Guide to Conducting a Systematic Literature Review of Information Systems Research," *SSRN Electronic Journal*, 2010. [Online]. Available: http://dx.doi.org/10.2139/ssrn.1954824

[80] Y. Xiao and M. Watson, "Guidance on Conducting a Systematic Literature Review," *Journal of Planning Education and Research*, vol. 39, no. 1, pp. 93–112, 2019. [Online]. Available: https://doi.org/10.1177/0739456X17723971

[81] M. J. Page, J. E. McKenzie, P. M. Bossuyt, I. Boutron, T. C. Hoffmann, C. D. Mulrow, L. Shamseer, J. M. Tetzlaff, E. A. Akl, S. E. Brennan *et al.*, "The PRISMA 2020 statement: an updated guideline for reporting systematic reviews," *Systematic Reviews*, vol. 10, no. 1, pp. 1–11, 2021. [Online]. Available: https://doi.org/10.1186/s13643-021-01626-4

[82] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering – A systematic literature review," *Information and Software Technology*, vol. 51, no. 1, pp. 7–15, Jan 2009. [Online]. Available: https://doi.org/10.1016%2Fj.infsof.2008.09.009

[83] H. Owen, J. Zarrin, and S. M. Pour, "A Survey on Botnets, Issues, Threats, Methods, Detection and Prevention," *Journal of Cybersecurity and Privacy*, vol. 2, no. 1, pp. 74–88, Feb 2022. [Online]. Available: https://doi.org/10.3390/jcp2010006

[84] M. Orabi, D. Mouheb, Z. Al Aghbari, and I. Kamel, "Detection of Bots in Social Media: A Systematic Review," *Information Processing and Management*, vol. 57, no. 4, p. 102250, 2020. [Online]. Available: https://doi.org/10.1016/j.ipm.2020.102250

[85] I. Mbona and J. H. Eloff, "Taxonomy of bots from a Cybersecurity perspective," in *Proceedings of the 21st International Conference on Security and Management (SAM 22)*, 2022. [Online]. Available: https://www.american-cse.org/static/CSCE22-book-abstracts-printing.pdf

[86] S. van der Linden, J. Roozenbeek, and J. Compton, "Inoculating Against Fake News About COVID-19," *Frontiers in Psychology*, vol. 11, pp. 1–7, 2020. [Online]. Available: https://doi.org/10.3389/fpsyg.2020.566790

[87] L. Vitkova, M. Kolomeec, and A. Chechulin, "Taxonomy and Bot Threats in Social Networks," in *2022 International Russian Automation Conference (RusAutoCon)*. IEEE, Sep. 2022. [Online]. Available: http://dx.doi.org/10.1109/rusautocon54946.2022.9896268

[88] J. Yang, Y.-L. Chen, L. Y. Por, and C. S. Ku, "A Systematic Literature Review of Information Security in Chatbots," *Applied Sciences*, vol. 13, no. 11, p. 6355, May 2023. [Online]. Available: http://dx.doi.org/10.3390/app13116355

[89] J. K. Haner and R. K. Knake, "Breaking botnets: A quantitative analysis of individual, technical, isolationist, and multilateral approaches to cybersecurity," *Journal of Cybersecurity*, vol. 7, no. 1, pp. 1–15, 2021. [Online]. Available: https://doi.org/10.1093/cybsec/tyab003

[90] A. Wang, W. Chang, S. Chen, and A. Mohaisen, "Delving into internet DDoS attacks by botnets: Characterization and analysis," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2843–2855, 2018. [Online]. Available: https://doi.org/10.1109/TNET.2018.2874896

[91] I. Ghafir, V. Prenosil, M. Hammoudeh, T. Baker, S. Jabbar, S. Khalid, and S. Jaf, "BotDet: A System for Real Time Botnet Command and Control Traffic Detection," *IEEE Access*, vol. 6, pp. 38 947–38 958, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2846740

[92] S. Saad, I. Traore, A. Ghorbani, B. Sayed, D. Zhao, W. Lu, J. Felix, and P. Hakimian, "Detecting P2P botnets through network behavior analysis and machine learning," *2011 9th Annual International Conference on Privacy, Security and Trust, PST 2011*, pp. 174–180, 2011. [Online]. Available: https://doi.org/10.1109/PST.2011.5971980

[93] H. Dhayal and J. Kumar, "Botnet and P2P Botnet Detection Strategies: A Review," in *2018 International Conference on Communication and Signal Processing (ICCSP)*. IEEE, Apr. 2018. [Online]. Available: http://dx.doi.org/10.1109/iccsp.2018.8524529

[94] M. Eslahi, R. Salleh, and N. B. Anuar, "Bots and botnets: An overview of characteristics, detection and challenges," *Proceedings - 2012 IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2012*, pp. 349–354, 2012. [Online]. Available: https://doi.org/10.1109/ICCSCE.2012.6487169

[95] D. Doran and S. S. Gokhale, "Web robot detection techniques: Overview and limitations," *Data Mining and Knowledge Discovery*, vol. 22, no. 1-2, pp. 183–210, 2011. [Online]. Available: https://doi.org/10.1007/s10618-010-0180-z

[96] S. Rovetta, G. Suchacka, and F. Masulli, "Bot recognition in a Web store: An approach based on unsupervised learning," *Journal of Network and Computer Applications*, vol. 157, May 2020. [Online]. Available: https://doi.org/10.1016/j.jnca.2020.102577

[97] B. Zhao, "Web scraping," *Encyclopedia of Big Data*, vol. 1, pp. 1–3, 2017. [Online]. Available: https://doi.org/10.1007/978-3-319-32001-4

[98] V. Subrahmanian, A. Azaria, S. Durst, V. Kagan, A. Galstyan, K. Lerman, L. Zhu, E. Ferrara, A. Flammini, and F. Menczer, "The DARPA Twitter Bot Challenge," *Computer*, vol. 49, no. 6, pp. 38–46, Jun 2016. [Online]. Available: https://doi.org/10.1109%2Fmc.2016.183

[99] E. Ferrara, O. Varol, C. Davis, F. Menczer, and A. Flammini, "The rise of social bots," *Communications of the ACM*, vol. 59, no. 7, pp. 96–104, Jun 2016. [Online]. Available: https://doi.org/10.1145%2F2818717

[100] A. R. Kang, S. H. Jeong, A. Mohaisen, and H. K. Kim, "Multimodal game bot detection using user behavioral characteristics," *SpringerPlus*, vol. 5, no. 1, 2016. [Online]. Available: https://doi.org/10.1186/s40064-016-2122-8

[101] B. Soper and J. Musacchio, "A botnet detection game," *2014 52nd Annual Allerton Conference on Communication, Control, and Computing, Allerton 2014*, pp. 294–303, 2014. [Online]. Available: https://doi.org/10.1109/ALLERTON.2014.7028469

[102] P. B. Brandtzaeg and A. Følstad, "Why people use chatbots," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10673 LNCS, pp. 377–392, 2017. [Online]. Available: https://doi.org/10.1007/978-3-319-70284-1_30

[103] E. Adamopoulou and L. Moussiades, "Chatbots: History, technology, and applications," *Machine Learning with Applications*, vol. 2, p. 100006, Dec 2020. [Online]. Available: https://doi.org/10.1016/j.mlwa.2020.100006

[104] W. Huang, K. F. Hew, and L. K. Fryer, "Chatbots for language learning—Are they really useful? A systematic review of chatbot-supported language learning," *Journal of*

Computer Assisted Learning, vol. 38, no. 1, pp. 237–257, Sep 2021. [Online]. Available: https://doi.org/10.1111%2Fjcal.12610

[105] M. Sahin, S. Agency, M. Relieu, T. Paristech, A. Francillon, S. Clara, M. Sahin, S. Antipolis, S. Antipolis, and A. Francillon, "Using chatbots against voice spam : Analyzing Lenny's effectiveness," *Thirteenth Symposium on Usable Privacy and Security (SOUPS 2017).*, vol. 13, no. 13, pp. 319–337, 2017. [Online]. Available: https://www.usenix.org/conference/soups2017/technical-sessions/presentation/sahin

[106] P. Wang, R. Angarita, and I. Renna, "Is this the Era of Misinformation yet: Combining Social Bots and Fake News to Deceive the Masses," *The Web Conference 2018 - Companion of the World Wide Web Conference, WWW 2018*, pp. 1557–1561, 2018. [Online]. Available: https://doi.org/10.1145/3184558.3191610

[107] K. Lee, J. Caverlee, and S. Webb, "Uncovering social spammers," in *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*. ACM, Jul 2010. [Online]. Available: https://doi.org/10.1145%2F1835449.1835522

[108] L. Kyumin, C. James, and W. Steve, "The social honeypot project: protecting online communities from spammers," *Proceedings of the 19th international conference on World wide web*, vol. 1, p. 1139, 2010. [Online]. Available: https://doi.org/10.1145/1772690.1772843

[109] Z. Chu, I. Widjaja, and H. Wang, "Detecting social spam campaigns on Twitter," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7341 LNCS, pp. 455–472, 2012. [Online]. Available: https://doi.org/10.1007/978-3-642-31284-7_27

[110] S. Cresci, F. Lillo, D. Regoli, S. Tardelli, and M. Tesconi, "$FAKE: evidence of spam and bot activity in stock microblogs on twitter," *12th International AAAI Conference on Web and Social Media, ICWSM 2018*, vol. 1, no. Icwsm, pp. 580–583, 2018. [Online]. Available: https://doi.org/10.1609/icwsm.v12i1.15073

[111] O. Çtlak, M. Dörterler, and . A. Doru, "A survey on detecting spam accounts on Twitter network," *Social Network Analysis and Mining*, vol. 9, no. 1, pp. 1–13, 2019. [Online]. Available: https://doi.org/10.1007/s13278-019-0582-x

[112] H. Kang, K. Wang, D. Soukal, F. Behr, and Z. Zheng, "Large-scale bot detection for search engines," *Proceedings of the 19th International Conference on World Wide Web, WWW '10*, pp. 501–510, 2010. [Online]. Available: https://doi.org/10.1145/1772690.1772742

[113] M. S. Iqbal, M. Zulkernine, F. Jaafar, and Y. Gu, "FCFraud: Fighting Click-Fraud from the User Side," *Proceedings of IEEE International Symposium on High Assurance Systems Engineering*, pp. 157–164, Jan 2016. [Online]. Available: https://doi.org/10.1109/HASE.2016.17

[114] G. S. Thejas, K. G. Boroojeni, K. Chandna, I. Bhatia, S. S. Iyengar, and N. R. Sunitha, "Deep learning-based model to fight against Ad click fraud," *ACMSE 2019 - Proceedings of the 2019 ACM Southeast Conference*, pp. 176–181, 2019. [Online]. Available: https://doi.org/10.1145/3299815.3314453

[115] S. Lim, J. Ha, H. Kim, Y. Kim, and S. Yang, "A SDN-oriented DDoS blocking scheme for botnet-based attacks," in *2014 Sixth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, Jul 2014. [Online]. Available: https://doi.org/10.1109%2Ficufn.2014.6876752

[116] C. Zhang and R. Green, "Communication security in internet of thing: Preventive measure and avoid DDoS attack over IoT network," *Simulation Series*, vol. 47, no. 3, pp. 8–15, 2015.

[117] S. Cresci, "A decade of social bot detection," *Communications of the ACM*, vol. 63, no. 10, pp. 72–83, Sep 2020. [Online]. Available: https://doi.org/10.1145%2F3409116

[118] J. Danaher, "Robot Betrayal: a guide to the ethics of robotic deception," *Ethics and Information Technology*, vol. 22, no. 2, pp. 117–128, 2020. [Online]. Available: https://doi.org/10.1007/s10676-019-09520-3

[119] W. Ma, P. Duan, S. Liu, G. Gu, and J.-C. Liu, "Shadow attacks: Automatically evading system-call-behavior based malware detection," *Journal in Computer Virology*, vol. 8, no. 1-2, pp. 1–13, 2012. [Online]. Available: https://doi.org/10.1007/s11416-011-0157-5

[120] H. Zhang, D. Yao, N. Ramakrishnan, and Z. Zhang, "Causality reasoning about network events for detecting stealthy malware activities," *Computers & Security*, vol. 58, no. May 2012, pp. 180–198, 2016. [Online]. Available: https://doi.org/10.1016/j.cose.2016.01.002

[121] Y. Ye, T. Li, D. Adjeroh, and S. S. Iyengar, "A survey on malware detection using data mining techniques," *ACM Computing Surveys*, vol. 50, no. 3, 2017. [Online]. Available: https://doi.org/10.1145/3073559

[122] A. Firdaus, N. B. Anuar, A. Karim, and M. F. A. Razak, "Discovering optimal features using static analysis and a genetic search based method for Android malware detection," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 6, pp. 712–736, Jun 2018. [Online]. Available: https://doi.org/10.1631%2Ffitee.1601491

[123] R. Upadhyaya and A. Jain, "Cyber ethics and cyber crime: A deep dwelved study into legality, ransomware, underground web and bitcoin wallet," in *2016 International Conference on Computing, Communication and Automation (ICCCA)*. IEEE, Apr 2016. [Online]. Available: https://doi.org/10.1109%2Fccaa.2016.7813706

[124] G. Cusack, O. Michel, and E. Keller, "Machine learning-based detection of ransomware using SDN," in *Proceedings of the 2018 ACM international workshop on security in software defined networks & network function virtualization*, 2018, pp. 1–6. [Online]. Available: https://doi.org/10.1145/3180465.3180467

[125] F. Mercaldo and A. Santone, "Deep learning for image-based mobile malware detection," *Journal of Computer Virology and Hacking Techniques*, vol. 16, no. 2, pp. 157–171, 2020. [Online]. Available: https://doi.org/10.1007/s11416-019-00346-7

[126] M. Shafahi, L. Kempers, and H. Afsarmanesh, "Phishing through social bots on Twitter," *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*, pp. 3703–3712, 2016. [Online]. Available: https://doi.org/10.1109/BigData.2016.7841038

[127] B. B. Gupta, A. Tewari, A. K. Jain, and D. P. Agrawal, "Fighting against phishing attacks: state of the art and future challenges," *Neural Computing and Applications*, vol. 28, no. 12, pp. 3629–3654, 2017. [Online]. Available: https://doi.org/10.1007/s00521-016-2275-y

[128] F. C. Akyon and M. Esat Kalfaoglu, "Instagram Fake and Automated Account Detection," *Proceedings - 2019 Innovations in Intelligent Systems and Applications Conference, ASYU 2019*, 2019. [Online]. Available: https://doi.org/10.1109/ASYU48272.2019.8946437

[129] A. Badawy, K. Lerman, and E. Ferrara, "Who falls for online political manipulation?" *The Web Conference 2019 - Companion of the World Wide Web Conference, WWW 2019*, pp. 162–168, 2019. [Online]. Available: https://doi.org/10.1145/3308560.3316494

[130] M. Al-Qurishi, S. M. M. Rahman, M. S. Hossain, A. Almogren, M. Alrubaian, A. Alamri, M. Al-Rakhami, and B. Gupta, "An efficient key agreement protocol for Sybil-precaution in online social networks," *Future Generation Computer Systems*, vol. 84, pp. 139–148, Jul 2018. [Online]. Available: https://doi.org/10.1016%2Fj.future.2017.07.055

[131] B. B. Zhu, J. Yan, Q. Li, C. Yang, J. Liu, N. Xu, M. Yi, and K. Cai, "Attacks and design of image recognition CAPTCHAs," *Proceedings of the ACM Conference on Computer and Communications Security*, pp. 187–200, 2010. [Online]. Available: https://doi.org/10.1145/1866307.1866329

[132] R. Lin, G. B. Bell, Y.-k. Lee, and T.-m. Rd, "A new CAPTCHA interface design for mobile devices," *Proceedings of the Twelfth Australasian User Interface Conference*, vol. 11, pp. 17–20, 2011.

[133] Q. Ye, Y. Chen, and B. Zhu, "The robustness of a new 3D CAPTCHA," *Proceedings - 11th IAPR International Workshop on Document Analysis Systems, DAS 2014*, pp. 319–323, 2014. [Online]. Available: https://doi.org/10.1109/DAS.2014.31

[134] H. Gao, X. Wang, F. Cao, Z. Zhang, L. Lei, J. Qi, and X. Liu, "Robustness of text-based completely automated public turing test to tell computers and humans apart," *IET Information Security*, vol. 10, no. 1, pp. 45–52, Jan 2016. [Online]. Available: https://doi.org/10.1049%2Fiet-ifs.2014.0381

[135] A. Khraisat, I. Gondal, P. Vamplew, J. Kamruzzaman, and A. Alazab, "A novel ensemble of hybrid intrusion detection system for detecting internet of things attacks," *Electronics (Switzerland)*, vol. 8, no. 11, 2019. [Online]. Available: https://doi.org/10.3390/electronics8111210

[136] A. Hernandez-Suarez, G. Sanchez-Perez, K. Toscano-Medina, V. Martinez-Hernandez, V. Sanchez, and H. Perez-Meana, "A web scraping methodology for bypassing twitter API restrictions," *arXiv preprint arXiv:1803.09875*, 2018. [Online]. Available: https://doi.org/10.48550/arXiv.1803.09875

[137] H. S. Dutta, K. Aggarwal, and T. Chakraborty, "DECIFE: Detecting Collusive Users Involved in Blackmarket following Services on Twitter," *HT 2021 - Proceedings of the*

32nd ACM Conference on Hypertext and Social Media, vol. 1, no. 1, pp. 91–100, Aug 2021. [Online]. Available: https://doi.org/10.1145/3465336.3475108

[138] J. Hinds, E. J. Williams, and A. N. Joinson, "It wouldn't happen to me: Privacy concerns and perspectives following the Cambridge Analytica scandal," *International Journal of Human Computer Studies*, vol. 143, p. 102498, June 2020. [Online]. Available: https://doi.org/10.1016/j.ijhcs.2020.102498

[139] P. Shi, Z. Zhang, and K.-K. R. Choo, "Detecting Malicious Social Bots Based on Clickstream Sequences," *IEEE Access*, vol. 7, pp. 28 855–28 862, 2019. [Online]. Available: https://doi.org/10.1109%2Faccess.2019.2901864

[140] Q. Li, H. Huang, R. Li, J. Lv, Z. Yuan, L. Ma, Y. Han, and Y. Jiang, "A comprehensive survey on DDoS defense systems: New trends and challenges," *Computer Networks*, vol. 233, p. 109895, Sep. 2023. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2023.109895

[141] E. van der Walt, J. H. Eloff, and J. Grobler, "Cyber-security: Identity deception detection on social media platforms," *Computers & Security*, vol. 78, pp. 76–89, 2018. [Online]. Available: https://doi.org/10.1016/j.cose.2018.05.015

[142] F. Salahdine and N. Kaabouch, "Social engineering attacks: A survey," *Future Internet*, vol. 11, 2019. [Online]. Available: https://doi.org/10.3390/FI11040089

[143] K. Chetioui, B. Bah, A. O. Alami, and A. Bahnasse, "Overview of Social Engineering Attacks on Social Networks," in *Procedia Computer Science*, vol. 198. Elsevier B.V., 2021, pp. 656–661. [Online]. Available: https://doi.org/10.1016/j.procs.2021.12.302

[144] Y. C. Chen and S. F. Wu, "FakeBuster: A Robust Fake Account Detection by Activity Analysis," *Proceedings - International Symposium on Parallel Architectures, Algorithms and Programming, PAAP*, pp. 108–110, 2018. [Online]. Available: https://doi.org/10.1109/PAAP.2018.00026

[145] J. Paavola, T. Helo, H. Jalonen, M. Sartonen, and A. M. Huhtinen, "The automated detection of trolling bots and cyborgs and the analysis of their impact in the social media," *European Conference on Information Warfare and Security, ECCWS*, no. 4, pp. 237–244, 2016.

[146] I. Alsmadi and M. J. O'Brien, "How Many Bots in Russian Troll Tweets?" *Information Processing & Management*, vol. 57, no. 6, p. 102303, Nov 2020. [Online]. Available: https://doi.org/10.1016%2Fj.ipm.2020.102303

[147] M. Al-Qurishi, M. Al-Rakhami, A. Alamri, M. Alrubaian, S. M. M. Rahman, and M. S. Hossain, "Sybil Defense Techniques in Online Social Networks: A Survey," *IEEE Access*, vol. 5, pp. 1200–1219, 2017. [Online]. Available: https://doi.org/10.1109%2Faccess.2017.2656635

[148] P. Roy and M. Sood, "Implementation of ensemble-based prediction model for detecting sybil accounts in an osn," in *Advances in Intelligent Systems and Computing*, vol. 1166. Springer, 2021, pp. 709–723. [Online]. Available: https://doi.org/10.1007/978-981-15-5148-2_62

[149] X. Xu, L. Liu, and B. Li, "A survey of CAPTCHA technologies to distinguish between human and computer," *Neurocomputing*, vol. 408, pp. 292–307, Sep 2020. [Online]. Available: https://doi.org/10.1016%2Fj.neucom.2019.08.109

[150] Q. Zhou and D. Pezaros, "Evaluation of Machine Learning Classifiers for Zero-Day Intrusion Detection – An Analysis on CIC-AWS-2018 dataset," *arXiv preprint arXiv:1905.03685*, 2019. [Online]. Available: http://arxiv.org/abs/1905.03685

[151] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for internet of things in smart city," *Future Generation Computer Systems*, vol. 107, pp. 433–442, 2020. [Online]. Available: https://doi.org/10.1016/j.future.2020.02.017

[152] C. Yin, Y. Zhu, J. Fei, and X. He, "A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks," *IEEE Access*, vol. 5, pp. 21 954–21 961, 2017. [Online]. Available: https://doi.org/10.1109/ACCESS.2017.2762418

[153] Y. Zhang, H. Gao, G. Pei, S. Luo, G. Chang, and N. Cheng, "A Survey of Research on CAPTCHA Designing and Breaking Techniques," in *2019 18th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, Aug. 2019. [Online]. Available: http://dx.doi.org/10.1109/trustcom/bigdatase.2019.00020

[154] J. S. Bhatia, R. K. Sehgal, and S. Kumar, *Honeynet Based Botnet Detection Using Command Signatures*. Springer Berlin Heidelberg, 2011, p. 6978. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-21153-9_7

[155] M. Al-Qurishi, M. S. Hossain, M. Alrubaian, S. M. M. Rahman, and A. Alamri, "Leveraging analysis of user behavior to identify malicious activities in large-scale social networks," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 2, pp. 799–813, 2018. [Online]. Available: https://doi.org/10.1109/TII.2017.2753202

[156] G. Sagirlar, B. Carminati, and E. Ferrari, "AutoBotCatcher: Blockchain-Based P2P Botnet Detection for the Internet of Things," in *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*. IEEE, Oct. 2018. [Online]. Available: http://dx.doi.org/10.1109/cic.2018.00-46

[157] F. Tegeler, X. Fu, G. Vigna, and C. Kruegel, "BotFinder: Finding bots in network traffic without deep packet inspection," *CoNEXT 2012 - Proceedings of the 2012 ACM Conference on Emerging Networking Experiments and Technologies*, pp. 349–360, 2012. [Online]. Available: https://doi.org/10.1145/2413176.2413217

[158] L. Durante, L. Seno, and A. Valenzano, "A formal model and technique to redistribute the packet filtering load in multiple firewall networks," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2637–2651, 2021. [Online]. Available: https://doi.org/10.1109/TIFS.2021.3057552

[159] C. Togay, A. Kasif, C. Catal, and B. Tekinerdogan, "A Firewall Policy Anomaly Detection Framework for Reliable Network Security," *IEEE Transactions on Reliability*, vol. 71, no. 1, pp. 339–347, 2022. [Online]. Available: https://doi.org/10.1109/TR.2021.3089511

[160] Z. Chiba, N. Abghour, K. Moussaid, A. El Omri, and M. Rida, "A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection," *Computers & Security*, vol. 75, pp. 36–58, 2018. [Online]. Available: https://doi.org/10.1016/j.cose.2018.01.023

[161] P. Dahiya and D. K. Srivastava, "Network Intrusion Detection in Big Dataset Using Spark," *Procedia Computer Science*, vol. 132, pp. 253–262, 2018. [Online]. Available: https://doi.org/10.1016/j.procs.2018.05.169

[162] A. Thakkar and R. Lohiya, "A Review of the Advancement in Intrusion Detection Datasets," *Procedia Computer Science*, vol. 167, no. 2019, pp. 636–645, 2020. [Online]. Available: https://doi.org/10.1016/j.procs.2020.03.330

[163] R. Samrin and D. Vasumathi, "Review on anomaly based network intrusion detection system," *International Conference on Electrical, Electronics, Communication Computer Technologies and Optimization Techniques, ICEECCOT 2017*, vol. 2018-Janua, pp. 141–147, 2018. [Online]. Available: https://doi.org/10.1109/ICEECCOT.2017.8284655

[164] A. L. Buczak and E. Guven, "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 2, pp. 1153–1176, Apr 2016. [Online]. Available: https://doi.org/10.1109/COMST.2015.2494502

[165] J. J. Tsai and Z. Yu, *Intrusion detection: a machine learning approach*. World Scientific, 2011, vol. 3.

[166] A. James, "Computer security threat monitoring and surveillance," *Technical report, James P. Anderson Company*, vol. 17, 1980. [Online]. Available: https://ci.nii.ac.jp/naid/10014688737/

[167] P. Vahdani Amoli and V. Amoli, "Unsupervised network intrusion detection systems for zero-day fast-spreading network attacks and botnets," *Jyväskylä studies in computing*, vol. 10, no. 231, pp. 1–13, 2015. [Online]. Available: http://urn.fi/URN:ISBN:978-951-39-6452-8

[168] N. H. Duong and H. D. Hai, "A semi-supervised model for network traffic anomaly detection," in *2015 17th International Conference on Advanced Communication Technology (ICACT)*. IEEE, Jul 2015. [Online]. Available: https://doi.org/10.1109%2Ficact.2015.7224759

[169] S. C. Pallaprolu, R. Sankineni, M. Thevar, G. Karabatis, and J. Wang, "Zero-Day Attack Identification in Streaming Data Using Semantics and Spark," *Proceedings - 2017 IEEE 6th International Congress on Big Data, BigData Congress 2017*, pp. 121–128, 2017. [Online]. Available: https://doi.org/10.1109/BigDataCongress.2017.25

[170] M. Zhu and N. Guo, "Abnormal Network Traffic Detection Based on Semi-Supervised Machine Learning," *DEStech Transactions on Engineering and Technology*

Research, vol. 1, no. ecame, pp. 628–635, 2018. [Online]. Available: https://doi.org/10.12783/dtetr/ecame2017/18466

[171] R. C. Aygun and A. G. Yavuz, "Network Anomaly Detection with Stochastically Improved Autoencoder Based Models," in *Proceedings - 4th IEEE International Conference on Cyber Security and Cloud Computing, CSCloud 2017 and 3rd IEEE International Conference of Scalable and Smart Cloud, SSC 2017*. Institute of Electrical and Electronics Engineers Inc., Jul 2017, pp. 193–198. [Online]. Available: https://doi.org/10.1109/CSCloud.2017.39

[172] K. A. Taher, B. Mohammed Yasin Jisan, and M. M. Rahman, "Network Intrusion Detection using Supervised Machine Learning Technique with Feature Selection," in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*. IEEE, Jan. 2019. [Online]. Available: http://dx.doi.org/10.1109/icrest.2019.8644161

[173] A. Blaise, M. Bouet, V. Conan, and S. Secci, "Detection of zero-day attacks: An unsupervised port-based approach," *Computer Networks*, vol. 180, p. 107391, Oct 2020. [Online]. Available: https://doi.org/10.1016%2Fj.comnet.2020.107391

[174] D. Xu and Y. Tian, "A Comprehensive Survey of Clustering Algorithms," *Annals of Data Science*, vol. 2, no. 2, pp. 165–193, 2015. [Online]. Available: https://doi.org/10.1007/s40745-015-0040-1

[175] S. Zavrak and M. Iskefiyeli, "Anomaly-Based Intrusion Detection from Network Flow Features Using Variational Autoencoder," *IEEE Access*, vol. 8, pp. 108 346–108 358, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.3001350

[176] H. Hindy, R. Atkinson, C. Tachtatzis, J. N. Colin, E. Bayne, and X. Bellekens, "Utilising deep learning techniques for effective zero-day attack detection," *Electronics (Switzerland)*, vol. 9, no. 10, pp. 1–16, 2020. [Online]. Available: https://doi.org/10.3390/electronics9101684

[177] G. Pu, L. Wang, J. Shen, and F. Dong, "A hybrid unsupervised clustering-based anomaly detection method," *Tsinghua Science and Technology*, vol. 26, no. 2, pp. 146–153, 2021. [Online]. Available: https://doi.org/10.26599/TST.2019.9010051

[178] T. Zoppi, A. Ceccarelli, and A. Bondavalli, "Unsupervised Algorithms to Detect Zero-Day Attacks: Strategy and Application," *IEEE Access*, vol. 9, pp. 90 603–90 615, 2021. [Online]. Available: https://doi.org/10.1109/access.2021.3090957

[179] Z. Tommaso, C. Andrea, and B. Andrea, "Evaluation of Anomaly Detection Algorithms Made Easy with RELOAD," in *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*, 2019, pp. 446–455. [Online]. Available: https://doi.org/10.1109/ISSRE.2019.00051

[180] N. Abdalgawad, A. Sajun, Y. Kaddoura, I. A. Zualkernan, and F. Aloul, "Generative Deep Learning to Detect Cyberattacks for the IoT-23 Dataset," *IEEE Access*, vol. 10, pp. 6430–6441, 2022. [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3140015

[181] P. Vuttipittayamongkol, E. Elyan, and A. Petrovski, "On the class overlap problem in imbalanced data classification," *Knowledge-Based Systems*, vol. 212, p. 106631, 2021. [Online]. Available: https://doi.org/10.1016/j.knosys.2020.106631

[182] L. Arshadi and A. H. Jahangir, "Benford's law behavior of internet traffic," *Journal of Network and Computer Applications*, vol. 40, no. 1, pp. 194–205, 2014. [Online]. Available: https://doi.org/10.1016/j.jnca.2013.09.007

[183] A. Laleh and J. A. Hossein, "An empirical study on TCP flow interarrival time distribution for normal and anomalous traffic," *International Journal of Communication Systems*, vol. 30, no. 1, 2017. [Online]. Available: https://doi.org/10.1002/dac.2881

[184] A. Iorliam, S. Tirunagari, A. T. S. Ho, S. Li, A. Waller, and N. Poh, ""Flow Size Difference" Can Make a Difference: Detecting Malicious TCP Network Flows Based on Benford's Law," *arXiv preprint arXiv:1609.04214*, pp. 1–14, 2016. [Online]. Available: http://arxiv.org/abs/1609.04214

[185] A. Iorliam, "Natural Laws (Benford's Law and Zipf's Law) for Network Traffic Analysis," *SpringerBriefs in Cybersecurity. Springer, Cham*, pp. 3–22, 2019. [Online]. Available: https://doi.org/10.1007/978-3-030-15210-9_2

[186] K. Sethi, R. Kumar, N. Prajapati, and P. Bera, "A Lightweight Intrusion Detection System using Benford's Law and Network Flow Size Difference," *2020 International Conference on COMmunication Systems and NETworkS, COMSNETS 2020*, vol. 10, pp. 1–6, 2020. [Online]. Available: https://doi.org/10.1109/COMSNETS48256.2020.9027422

[187] L. Sun, T. S. Anthony, H. Z. Xia, J. Chen, X. Huang, and Y. Zhang, "Detection and classification of malicious patterns in network traffic using Benfords law," in *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, Dec 2017. [Online]. Available: http://dx.doi.org/10.1109/apsipa.2017.8282154

[188] R. Ahmad, I. Alsmadi, W. Alhamdani, and L. Tawalbeh, "Zero-day attack detection: a systematic literature review," *Artificial Intelligence Review*, vol. 56, no. 10, pp. 10 733–10 811, Feb 2023. [Online]. Available: https://doi.org/10.1007%2Fs10462-023-10437-z

[189] I. Mbona and J. H. Eloff, "Feature selection using Benford's law to support detection of malicious social media bots," *Information Sciences*, vol. 582, pp. 369–381, Sep 2022. [Online]. Available: https://doi.org/10.1016/j.ins.2021.09.038

[190] D. Dukic, D. Keca, and D. Stipic, "Are you human? detecting bots on twitter using BERT," *Proceedings - 2020 IEEE 7th International Conference on Data Science and Advanced Analytics, DSAA 2020*, pp. 631–636, 2020. [Online]. Available: https://doi.org/10.1109/DSAA49011.2020.00089

[191] D. B. Kurka, A. Godoy, and F. J. Von Zuben, "Online Social Network Analysis: A Survey of Research Applications in Computer Science," *arXiv preprint arXiv:1504.05655*, pp. 1–55, 2015. [Online]. Available: http://arxiv.org/abs/1504.05655

[192] A. Talha and R. Kara, "A Survey of Spam Detection Methods on Twitter," *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 3, 2017. [Online]. Available: https://doi.org/10.14569%2Fijacsa.2017.080305

[193] M. R. Islam, S. Liu, X. Wang, and G. Xu, "Deep learning for misinformation detection on online social networks: a survey and new perspectives," *Social Network Analysis and Mining*, vol. 10, no. 1, pp. 1–20, 2020. [Online]. Available: https://doi.org/10.1007/s13278-020-00696-x

[194] M. A. Keijzer and M. Mäs, "The strength of weak bots," *Online Social Networks and Media*, vol. 21, p. 100106, 2021. [Online]. Available: https://doi.org/10.1016/j.osnem.2020.100106

[195] Y. Luo, M. Liu, J. Wang, C. Yuan, and T. Liu, "When Secure Data Sharing Meets Blockchain: Overview, Challenges and Future Prospects," in *The 2022 4th International*

Conference on Blockchain Technology. ACM, Mar 2022, pp. 1–8. [Online]. Available: https://doi.org/10.1145/3532640.3532641

[196] G. Mitra, P. K. Vairam, S. Saha, N. Chandrachoodan, and V. Kamakoti, "Snoopy: A Webpage Fingerprinting Framework with Finite Query Model for Mass-Surveillance," *arXiv preprint arXiv:2205.15037*, vol. 1, no. 1, pp. 1–19, May 2022. [Online]. Available: http://arxiv.org/abs/2205.15037

[197] Hitkul, O. Gurjar, A. Sadaria, K. Gupta, S. Srikanth, R. R. Shah, and P. Kumaraguru, "Are Bots Humans? Analysis of Bot Accounts in 2019 Indian Lok Sabha Elections (Workshop Paper)," *Proceedings - 2020 IEEE 6th International Conference on Multimedia Big Data, BigMM 2020*, pp. 441–450, 2020. [Online]. Available: https://doi.org/10.1109/BigMM50055.2020.00073

[198] X. Li, B. A. Azad, A. Rahmati, and N. Nikiforakis, "Good Bot, Bad Bot: Characterizing Automated Browsing Activity," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 1589–1605. [Online]. Available: https://doi.org/10.1109/SP40001.2021.00079

[199] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Who is tweeting on twitter: Human, bot, or cyborg?" *Proceedings - Annual Computer Security Applications Conference, ACSAC*, pp. 21–30, 2010. [Online]. Available: https://doi.org/10.1145/1920261.1920265

[200] C. Zi, G. Steven, W. Haining, and S. Jajodia, "Detecting automation of Twitter accounts: Are you a human, bot, or cyborg?" *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 6, pp. 811–824, 2012. [Online]. Available: https://doi.org/10.1109/TDSC.2012.75

[201] C. A. Davis, O. Varol, E. Ferrara, A. Flammini, and F. Menczer, "BotOrNot: A System to Evaluate Social Bots," in *Proceedings of the 25th International Conference Companion on World Wide Web - WWW 16 Companion*. ACM Press, 2016. [Online]. Available: http://dx.doi.org/10.1145/2872518.2889302

[202] O. Varol, E. Ferrara, C. Davis, F. Menczer, and A. Flammini, "Online Human-Bot Interactions: Detection, Estimation, and Characterization," *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 11, no. 1, p. 280289, May 2017. [Online]. Available: http://dx.doi.org/10.1609/icwsm.v11i1.14871

[203] E. Van Der Walt and J. H. Eloff, "Using Machine Learning to Detect Fake Identities: Bots vs Humans," *IEEE Access*, vol. 6, pp. 6540–6549, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2796018

[204] S. Cresci, A. Spognardi, M. Petrocchi, M. Tesconi, R. D. Pietro, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race," *26th International World Wide Web Conference 2017, WWW 2017 Companion*, pp. 963–972, 2017. [Online]. Available: https://doi.org/10.1145/3041021.3055135

[205] S. Khaled, N. El-Tazi, and H. M. Mokhtar, "Detecting Fake Accounts on Social Media," *Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018*, vol. 1, no. Mar, pp. 3672–3681, 2019. [Online]. Available: https://doi.org/10.1109/BigData.2018.8621913

[206] N. Pilnenskiy and I. Smetannikov, "Feature selection algorithms as one of the python data analytical tools," *Future Internet*, vol. 12, no. 3, 2020. [Online]. Available: https://doi.org/10.3390/fi12030054

[207] A. Zheng and A. Casari, *Feature engineering for machine learning: principles and techniques for data scientists*. O'Reilly Media, 2018.

[208] A. H. Wang, "Detecting spam bots in online social networking sites: A machine learning approach," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6166 LNCS, pp. 335–342, 2010. [Online]. Available: https://doi.org/10.1007/978-3-642-13739-6_25

[209] H. Ping and S. Qin, "A social bots detection model based on deep learning algorithm," *International Conference on Communication Technology Proceedings, ICCT*, vol. 2019-Octob, pp. 1435–1439, 2019. [Online]. Available: https://doi.org/10.1109/ICCT.2018.8600029

[210] J. Schnebly and S. Sengupta, "Random forest twitter bot classifier," *2019 IEEE 9th Annual Computing and Communication Workshop and Conference, CCWC 2019*, pp. 506–512, 2019. [Online]. Available: https://doi.org/10.1109/CCWC.2019.8666593

[211] C. Xiao, D. M. Freeman, and T. Hwa, "Detecting clusters of fake accounts in online social networks," in *Proceedings of the 8th ACM Workshop on Artificial*

Intelligence and Security, Oct 2015, pp. 91–101. [Online]. Available: http://dx.doi.org/10.1145/2808769.2808779

[212] Z. Gilani, R. Farahbakhsh, G. Tyson, L. Wang, and J. Crowcroft, "Of Bots and Humans (on Twitter)," in *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, Jul 2017. [Online]. Available: https://doi.org/10.1145%2F3110025.3110090

[213] Z. Gilani, R. Farahbakhsh, G. Tyson, and J. Crowcroft, "A Large-scale Behavioural Analysis of Bots and Humans on Twitter," *ACM Transactions on the Web*, vol. 13, no. 1, pp. 1–23, Feb 2019. [Online]. Available: https://doi.org/10.1145%2F3298789

[214] I. Kayes and A. Iamnitchi, "Privacy and security in online social networks: A survey," *Online Social Networks and Media*, vol. 3-4, pp. 1–21, 2017. [Online]. Available: https://doi.org/10.1016/j.osnem.2017.09.001

[215] J. Rodríguez-Ruiz, J. I. Mata-Sánchez, R. Monroy, O. Loyola-González, and A. López-Cuevas, "A one-class classification approach for bot detection on Twitter," *Computers & Security*, vol. 91, 2020. [Online]. Available: https://doi.org/10.1016/j.cose.2020.101715

[216] A. Dorri, M. Abadi, and M. Dadfarnia, "SocialBotHunter: Botnet detection in twitter-like social networking services using semi-supervised collective classification," *Proceedings - IEEE 16th International Conference on Dependable, Autonomic and Secure Computing, IEEE 16th International Conference on Pervasive Intelligence and Computing, IEEE 4th International Conference on Big Data Intelligence and Computing and IEEE 3*, pp. 496–503, 2018. [Online]. Available: https://doi.org/10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00097

[217] X. Zheng, Z. Zeng, Z. Chen, Y. Yu, and C. Rong, "Detecting spammers on social networks," *Neurocomputing*, vol. 159, pp. 27–34, Jul 2015. [Online]. Available: https://doi.org/10.1016%2Fj.neucom.2015.02.047

[218] N. Chavoshi, H. Hamooni, and A. Mueen, "DeBot: Twitter Bot Detection via Warped Correlation," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, Dec 2016. [Online]. Available: https://doi.org/10.1109%2Ficdm.2016.0096

[219] J. Golbeck, "Benford's law applies to online social networks," *PLoS ONE*, vol. 10, no. 8, Aug 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0135169

[220] J.Golbeck, "Benford's Law can detect malicious social bots," *First Monday*, Aug 2019. [Online]. Available: https://doi.org/10.5210%2Ffm.v24i8.10163

[221] D. Striga and V. Podobnik, "Benford's Law and Dunbar's Number: Does Facebook Have a Power to Change Natural and Anthropological Laws?" *IEEE Access*, vol. 6, pp. 14 629–14 642, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2805712

[222] S. Maurus and C. Plant, "Let's see your digits: Anomalous-state detection using Benford's Law," *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, vol. Part F1296, pp. 977–986, 2017. [Online]. Available: https://doi.org/10.1145/3097983.3098101

[223] I. T. Javed, K. Toumi, F. Alharbi, T. Margaria, and N. Crespi, "Detecting Nuisance Calls over Internet Telephony Using Caller Reputation," *Electronics*, vol. 10, no. 3, p. 353, Feb 2021. [Online]. Available: https://doi.org/10.3390%2Felectronics10030353

[224] M. Dekker and L. Alevizos, "A threatintelligence driven methodology to incorporate uncertainty in cyber risk analysis and enhance decisionmaking," *SECURITY AND PRIVACY*, vol. 7, no. 1, Jul. 2023. [Online]. Available: http://dx.doi.org/10.1002/spy2.333

[225] M. Noorizadeh, M. Shakerpour, N. Meskin, D. Unal, and K. Khorasani, "A Cyber-Security Methodology for a Cyber-Physical Industrial Control System Testbed," *IEEE Access*, vol. 9, p. 1623916253, 2021. [Online]. Available: http://dx.doi.org/10.1109/access.2021.3053135

[226] R. Coulter, Q.-L. Han, L. Pan, J. Zhang, and Y. Xiang, "Data-Driven Cyber Security in Perspective - Intelligent Traffic Analysis," *IEEE Transactions on Cybernetics*, vol. 50, no. 7, p. 30813093, Jul. 2020. [Online]. Available: http://dx.doi.org/10.1109/tcyb.2019.2940940

[227] G. Spanos, K. M. Giannoutakis, K. Votis, B. Viano, J. Augusto-Gonzalez, G. Aivatoglou, and D. Tzovaras, "A Lightweight Cyber-Security Defense Framework for Smart Homes," in *2020 International Conference on INnovations in Intelligent SysTems and Applications (INISTA)*. IEEE, Aug. 2020. [Online]. Available: http://dx.doi.org/10.1109/inista49547.2020.9194689

[228] S. K. Datta, "DRAFT - A Cybersecurity Framework for IoT Platforms," in *2020 Zooming Innovation in Consumer Technologies Conference (ZINC)*. IEEE, May 2020. [Online]. Available: http://dx.doi.org/10.1109/zinc50678.2020.9161441

[229] V. Parlapalli, V. Jayaram, S. G. Aarella, K. Peddireddy, and R. R. Palle, "Enhancing Cybersecurity: A Deep Dive into Augmented Intelligence Through Machine Learning and Image Processing," in *2023 International Workshop on Artificial Intelligence and Image Processing (IWAIIP)*. IEEE, Dec. 2023. [Online]. Available: http://dx.doi.org/10.1109/iwaiip58158.2023.10462845

[230] S. Magnani, R. Doriguzzi-Corin, and D. Siracusa, "Enhancing Network Intrusion Detection: An Online Methodology for Performance Analysis," in *2023 IEEE 9th International Conference on Network Softwarization (NetSoft)*. IEEE, Jun. 2023. [Online]. Available: http://dx.doi.org/10.1109/netsoft57336.2023.10175465

[231] L. Arnau Muoz, J. V. Bern Martnez, F. Maci Prez, and I. Lorenzo Fonseca, "Anomaly detection system for data quality assurance in IoT infrastructures based on machine learning," *Internet of Things*, vol. 25, p. 101095, Apr. 2024. [Online]. Available: http://dx.doi.org/10.1016/j.iot.2024.101095

[232] B. Horowitz, P. Beling, C. Fleming, S. Adams, B. Carter, T. Sherburne, C. Elks, G. Bakirtzis, F. Shull, and N. R. Mead, "Cyber security requirements methodology," *Systems Engineering Research Center, Technical Report SERC-2018-TR-110*, 2018.

[233] R. Kaushik and M. Dave, "Malware Detection System Using Ensemble Learning: Tested Using Synthetic Data," in *Data Engineering and Communication Technology: Proceedings of ICDECT 2021*, 2021, pp. 153–164. [Online]. Available: https://doi.org/10.1007/978-981-16-0081-4_16

[234] M. Woodard, S. S. Sarvestani, and A. R. Hurson, "Chapter Two - A Survey of Research on Data Corruption in CyberPhysical Critical Infrastructure Systems," in *Advances in Computers*, A. R. Hurson, Ed. Elsevier, 2015, vol. 98, pp. 59–87. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0065245815000285

[235] S. A. V. Jatti and V. K. Sontif, "Intrusion Detection Systems," *International Journal of Recent Technology and Engineering*, vol. 8, no. 2S11, pp. 3976–3983, Nov 2019. [Online]. Available: https://doi.org/10.35940%2Fijrte.b1540.0982s1119

[236] G. Bonaccorso, *Mastering Machine Learning Algorithms: Expert techniques for implementing popular machine learning algorithms, fine-tuning your models, and understanding how they work*. OReilly Media, 2020.

[237] J. Brownlee, *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*. Machine Learning Mastery, 2020.

[238] J. Cai, J. Luo, S. Wang, and S. Yang, "Feature selection in machine learning: A new perspective," *Neurocomputing*, vol. 300, pp. 70–79, 2018. [Online]. Available: https://doi.org/10.1016/j.neucom.2017.11.077

[239] A. Alhefdhi, H. K. Dam, H. Hata, and A. Ghose, "Generating pseudo-code from source code using deep learning," in *Proceedings - 25th Australasian Software Engineering Conference, ASWEC 2018*. Institute of Electrical and Electronics Engineers Inc., Dec 2018, pp. 21–25. [Online]. Available: https://doi.org/10.1109/ASWEC.2018.00011

[240] S. Rai and A. Gupta, "Generation of Pseudo Code from the Python Source Code using Rule-Based Machine Translation," *arXiv preprint arXiv:1906.06117*, Jun 2019. [Online]. Available: http://arxiv.org/abs/1906.06117

[241] H. B. Abdalla, "A brief survey on big data: technologies, terminologies and data-intensive applications," *Journal of Big Data*, vol. 9, no. 1, 2022. [Online]. Available: https://doi.org/10.1186/s40537-022-00659-3

[242] C. Zhang, J. Guo, and J. Lu, "Research on Classification Method of High-Dimensional Class-Imbalanced Data Sets Based on SVM," in *2017 IEEE Second International Conference on Data Science in Cyberspace (DSC)*. IEEE, Jun 2017. [Online]. Available: https://doi.org/10.1109%2Fdsc.2017.63

[243] M. S. Albarrak, M. Elnahass, S. Papagiannidis, and A. Salama, "The effect of twitter dissemination on cost of equity: A big data approach," *International Journal of Information Management*, vol. 50, pp. 1–16, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0268401218312921

[244] J. Miao and L. Niu, "A Survey on Feature Selection," *Procedia Computer Science*, vol. 91, pp. 919–926, 2016. [Online]. Available: https://doi.org/10.1016%2Fj.procs.2016.07.111

[245] F. Thabtah, S. Hammoud, F. Kamalov, and A. Gonsalves, "Data imbalance in classification: Experimental evaluation," *Information Sciences*, vol. 513, pp. 429–441, 2020. [Online]. Available: https://doi.org/10.1016/j.ins.2019.11.004

[246] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," *Proceedings of 2014 Science and Information Conference, SAI 2014*, vol. 1, pp. 372–378, 2014. [Online]. Available: https://doi.org/10.1109/SAI.2014.6918213

[247] S. García, J. Luengo, and F. Herrera, "Feature selection," *Intelligent Systems Reference Library*, vol. 72, no. 6, pp. 163–193, 2015. [Online]. Available: https://doi.org/10.1007/978-3-319-10247-4_7

[248] J. Gui, Z. Sun, S. Ji, D. Tao, and T. Tan, "Feature selection based on structured sparsity: a comprehensive study," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 7, pp. 1490–1507, 2017. [Online]. Available: https://doi.org/10.1109/TNNLS.2016.2551724

[249] M. A. Ferrag, L. Shu, O. Friha, and X. Yang, "Cyber Security Intrusion Detection for Agriculture 4.0: Machine Learning-Based Solutions, Datasets, and Future Directions," *IEEE/CAA Journal of Automatica Sinica*, vol. 9, 2022. [Online]. Available: https://doi.org/10.1109/JAS.2021.1004344

[250] J. K. F. Bowles, A. Silvina, E. Bin, and M. Vinov, "On Defining Rules for Cancer Data Fabrication," in *Rules and Reasoning: 4th International Joint Conference, RuleML+ RR 2020, Oslo, Norway, June 29–July 1, 2020, Proceedings 4*, 2020, pp. 168–176. [Online]. Available: https://www.ndc.scot.nhs.uk/National-Datasets/

[251] I. F. Kilincer, F. Ertam, and A. Sengur, "Machine learning methods for cyber security intrusion detection: Datasets and comparative study," *Computer Networks*, vol. 188, 2021. [Online]. Available: https://doi.org/10.1016/j.comnet.2021.107840

[252] A. Ferriyan, A. H. Thamrin, K. Takeda, and J. Murai, "Generating network intrusion detection dataset based on real and encrypted synthetic attack traffic," *Applied Sciences (Switzerland)*, vol. 11, 2021. [Online]. Available: https://doi.org/10.3390/app11177868

[253] F. K. Dankar and M. Ibrahim, "Fake it till you make it: Guidelines for effective synthetic data generation," *Applied Sciences (Switzerland)*, vol. 11, 2021. [Online]. Available: https://doi.org/10.3390/app11052158

[254] D. B. Rubin, "Statistical disclosure limitation," *Journal of official Statistics*, vol. 9, pp. 461–468, 1993. [Online]. Available: https://www.scb.se/contentassets/ca21efb41fee47d293bbee5bf7be7fb3/discussion-statistical-disclosure-limitation2.pdf

[255] J. Kim, D. Kim, S. H. Lee, and S. Chi, "Hybrid DNN training using both synthetic and real construction images to overcome training data shortage," *Automation in Construction*, vol. 149, 2023. [Online]. Available: https://doi.org/10.1016/j.autcon.2023.104771

[256] A. Figueira and B. Vaz, "Survey on Synthetic Data Generation, Evaluation Methods and GANs," *MDPI Mathematics*, vol. 10, 2022. [Online]. Available: https://doi.org/10.3390/math10152733

[257] V. Kumar and D. Sinha, "Synthetic attack data generation model applying generative adversarial network for intrusion detection," *Computers & Security*, vol. 125, 2023. [Online]. Available: https://doi.org/10.1016/j.cose.2022.103054

[258] R. Bhusal, B. Taner, and K. Subbarao, "On the Phase Margin of Networked Dynamical Systems and Fabricated Attacks of an Intruder," in *2020 American Control Conference (ACC)*, 2020, pp. 3279–3284. [Online]. Available: https://doi.org/10.23919/ACC45564.2020.9147500

[259] O. Yavanoglu and M. Aydos, "A review on cyber security datasets for machine learning algorithms," *Proceedings - 2017 IEEE International Conference on Big Data, Big Data 2017*, vol. 2018-Janua, no. December, pp. 2186–2193, 2017. [Online]. Available: https://doi.org/10.1109/BigData.2017.8258167

[260] I. Mbona and J. H. Eloff, "Detecting Zero-Day Intrusion Attacks Using Semi-Supervised Machine Learning Approaches," *IEEE Access*, vol. 10, no. Jul, pp. 69 822–69 838, 2022. [Online]. Available: https://doi.org/10.1109/ACCESS.2022.3187116

[261] W. J. Scheirer, L. P. Jain, and T. E. Boult, "Probability models for open set recognition," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, 2014. [Online]. Available: https://doi.org/10.1109/TPAMI.2014.2321392

[262] S. Kim, C. Hwang, and T. Lee, "Anomaly based unknown intrusion detection in endpoint environments," *Electronics (Switzerland)*, vol. 9, pp. 1–21, Jun 2020. [Online]. Available: https://doi.org/10.3390/electronics9061022

[263] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019. [Online]. Available: https://doi.org/10.1016/j.future.2019.05.041

[264] N. Koroniotis, "Designing an effective network forensic framework for the investigation of botnets in the Internet of Things," Ph.D. dissertation, UNSW Sydney, 2020. [Online]. Available: https://doi.org/10.26190/unsworks/21942

[265] S. K. Singh and P. K. Roy, "Detecting Malicious DNS over HTTPS Traffic Using Machine Learning," in *2020 International Conference on Innovation and Intelligence for Informatics, Computing and Technologies, 3ICT 2020*, 2020, pp. 1–6. [Online]. Available: https://doi.org/10.1109/3ICT51146.2020.9312004

[266] Y. Khodjaeva and N. Zincir-Heywood, "Network Flow Entropy for Identifying Malicious Behaviours in DNS Tunnels," in *Proceedings of the 16th International Conference on Availability, Reliability and Security*, 2021, pp. 1–7. [Online]. Available: https://doi.org/10.1145/3465481.3470089

[267] S. Al-Mashhadi, M. Anbar, S. Karuppayah, and A. K. Al-Ani, *A Review of Botnet Detection Approaches Based on DNS Traffic Analysis*. Springer Singapore, 2019, p. 305321. [Online]. Available: http://dx.doi.org/10.1007/978-981-13-6031-2_21

[268] C. Patsakis, F. Casino, and V. Katos, "Encrypted and covert DNS queries for botnets: Challenges and countermeasures," *Computers & Security*, vol. 88, p. 101614, Jan. 2020. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2019.101614

[269] M. Singh, M. Singh, and S. Kaur, "Issues and challenges in DNS based botnet detection: A survey," *Computers & Security*, vol. 86, p. 2852, Sep. 2019. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2019.05.019

[270] Z. S. Abdallah and G. I. Webb, "Data Preparation," *Encyclopedia of Machine Learning and Data Mining*, pp. 318–327, 2017. [Online]. Available: https://doi.org/10.1007/978-1-4899-7687-1

[271] V. N. Gudivada, A. Apon, and J. Ding, "Data Quality Considerations for Big Data and Machine Learning: Going Beyond Data Cleaning and Transformations," *International Journal on Advances in Software*, vol. 10, no. 1, pp. 1–20, 2017. [Online]. Available:

https://personales.upv.es/thinkmind/dl/journals/soft/soft_v10_n12_2017/soft_v10_n12_2017_1.pdf

[272] R. Mahanti, *Data Quality: Dimensions, Measurement, Strategy, Management, and Governance*. Quality Press, 2019.

[273] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *Journal of Computational Science*, vol. 25, pp. 152–160, 2018. [Online]. Available: https://doi.org/10.1016/j.jocs.2017.03.006

[274] Kurniabudi, D. Stiawan, Darmawijoyo, M. Y. B. Bin Idris, A. M. Bamhdi, and R. Budiarto, "CICIDS-2017 Dataset Feature Analysis with Information Gain for Anomaly Detection," *IEEE Access*, vol. 8, pp. 132 911–132 921, 2020. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.3009843

[275] H. Khalil, M. U. Khan, and M. Ali, "Feature Selection for Unsupervised Bot Detection," *2020 3rd International Conference on Computing, Mathematics and Engineering Technologies: Idea to Innovation for Building the Knowledge Economy, iCoMET 2020*, pp. 1–7, 2020. [Online]. Available: https://doi.org/10.1109/iCoMET48670.2020.9074131

[276] J. L. Leevy, J. Hancock, R. Zuech, and T. M. Khoshgoftaar, "Detecting cybersecurity attacks using different network features with LightGBM and XGBoost learners," *Proceedings - 2020 IEEE 2nd International Conference on Cognitive Machine Intelligence, CogMI 2020*, pp. 190–197, 2020. [Online]. Available: https://doi.org/10.1186/s40537-021-00426-w

[277] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers and Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014. [Online]. Available: https://doi.org/10.1016/j.compeleceng.2013.11.024

[278] J. K. Jaiswal and R. Samikannu, "Application of Random Forest Algorithm on Feature Subset Selection and Classification and Regression," in *2017 World Congress on Computing and Communication Technologies (WCCCT)*, 2017, pp. 65–68. [Online]. Available: https://doi.org/10.1109/WCCCT.2016.25

[279] A. N. Asadi, "An approach for detecting anomalies by assessing the inter-arrival time of UDP packets and flows using Benford's law," in *2015 2nd International Conference*

on Knowledge-Based Engineering and Innovation (KBEI). IEEE, Nov 2015. [Online]. Available: https://doi.org/10.1109%2Fkbei.2015.7436057

[280] K. L. Kurien and A. A. Chikkamannur, "Benford's Law and Deep Learning Autoencoders: An approach for Fraud Detection of Credit card Transactions in Social Media," in *2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*. IEEE, May 2019. [Online]. Available: https://doi.org/10.1109%2Frteict46194.2019.9016804

[281] A. Berger and T. Hill, *An introduction to Benford's law*. Princeton University Press, 2015.

[282] M. J. Nigrini, *Forensic Analytics: Methods and Techniques for Forensic Accounting Investigations*. John Wiley & Sons, 2012. [Online]. Available: https://doi.org/10.1002/9781118386798

[283] S. Newcomb, "Note on the Frequency of Use of the Different Digits in Natural Numbers," *American Journal of Mathematics*, vol. 4, no. 1/4, p. 39, 1881. [Online]. Available: https://doi.org/10.2307/2369148

[284] F. Benford, "The Law of Anomalous Numbers," *Proceedings of the American Philosophical Society*, vol. 78, no. 4, pp. 551–572, 1938. [Online]. Available: http://www.jstor.org/stable/984802

[285] R. S. Pinkham, "On the distribution of first significant digits," *The Annals of Mathematical Statistics*, vol. 32, no. 4, pp. 1223–1230, 1961. [Online]. Available: https://www.jstor.org/stable/2237922

[286] S. Afanasiev and A. Smirnova, "Predictive fraud analytics: B-tests," *Journal of Operational Risk*, vol. 13, no. 4, pp. 17–46, 2018. [Online]. Available: https://doi.org/10.21314/JOP.2018.213

[287] M. Kalameyets, D. Levshun, S. Soloviev, A. Chechulin, and I. Kotenko, "Social networks bot detection using Benford's law," in *13th International Conference on Security of Information and Networks*. ACM, Nov 2020. [Online]. Available: https://doi.org/10.1145%2F3433174.3433589

[288] M. J. Nigrini and L. J. Mittermaier, "The use of Benford's Law as an aid in analytical procedures," *Auditing*, vol. 16, no. 2, pp. 66–67, 1997. [Online]. Available:

https://openurl.ebsco.com/EPDB%3Agcd%3A13%3A1121741/detailv2?sid=ebsco%3Aplink%3Ascholar&id=ebsco%3Agcd%3A1302507&crl=c

[289] E. Badal-Valero, J. A. Alvarez-Jareño, and J. M. Pavía, "Combining Benford's Law and machine learning to detect money laundering. An actual Spanish court case," *Forensic Science International*, vol. 282, pp. 24–34, 2018. [Online]. Available: https://doi.org/10.1016/j.forsciint.2017.11.008

[290] L. Barabesi and L. Pratelli, "On the Generalized Benford law," *Statistics & Probability Letters*, vol. 160, p. 108702, May 2020. [Online]. Available: https://doi.org/10.1016%2Fj.spl.2020.108702

[291] E. Druic, B. Oancea, and C. Vâlsan, "Benford's law and the limits of digit analysis," *International Journal of Accounting Information Systems*, vol. 31, no. July, pp. 75–82, 2018. [Online]. Available: https://doi.org/10.1016/j.accinf.2018.09.004

[292] Z. Jasak and L. Banjanović-Mehmedovic, "Detecting anomalies by Benford's Law," *Proceedings of the 8th IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2008*, pp. 453–458, 2008. [Online]. Available: https://doi.org/10.1109/ISSPIT.2008.4775660

[293] E. P. Balanzario and J. Sánchez-Ortiz, "Sufficient conditions for Benford's law," *Statistics and Probability Letters*, vol. 80, no. 23-24, pp. 1713–1719, 2010. [Online]. Available: https://doi.org/10.1016/j.spl.2010.07.014

[294] I. Mbona and J. H. P. Eloff, "Classifying social media bots as malicious or benign using semi-supervised machine learning," *Journal of Cybersecurity*, vol. 9, no. 1, Jan. 2023. [Online]. Available: http://dx.doi.org/10.1093/cybsec/tyac015

[295] I. Mbona and J. Eloff, "Evaluating a semi-supervised intrusion detection algorithm through Benford's Law," in *Proceedings of the 17th International Conference on Data Science (ICDATA 2021)*, 2021. [Online]. Available: https://www.american-cse.org/static/CSCE21%20book%20abstracts.pdf

[296] J. Zieba-Palus, *Forensic analytics*. Handbook of Trace Analysis: Fundamentals and Applications, 2016. [Online]. Available: https://doi.org/10.1007/978-3-319-19614-5_11

[297] A. Singh, "Foundations of Machine Learning," 2019. [Online]. Available: https://doi.org/10.2139/ssrn.3399990

[298] A. Cholaquidis, R. Fraiman, and M. Sued, "On semi-supervised learning," *TEST*, vol. 29, no. 4, pp. 914–937, 2020. [Online]. Available: https://doi.org/10.1007/s11749-019-00690-2

[299] R. Michalski, J. Carbonell, and T. Mitchell, *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.

[300] P. Perera and V. M. Patel, "Learning Deep Features for One-Class Classification," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5450–5463, 2019. [Online]. Available: https://doi.org/10.1109/TIP.2019.2917862

[301] T. Amr, "Hands-On Machine Learning with scikit-learn and Scientific Python Toolkits," *OReilly Media*, p. 384, 2019.

[302] G. Fernandes, J. J. Rodrigues, L. F. Carvalho, J. F. Al-Muhtadi, and M. L. Proença, "A comprehensive survey on network anomaly detection," *Telecommunication Systems*, vol. 70, no. 3, pp. 447–489, 2019. [Online]. Available: https://doi.org/10.1007/s11235-018-0475-8

[303] K. Anand, J. Kumar, and K. Anand, "Anomaly detection in online social network: A survey," in *Proceedings of the International Conference on Inventive Communication and Computational Technologies, ICICCT 2017*, vol. 1, 2017, pp. 456–459. [Online]. Available: https://doi.org/10.1109/ICICCT.2017.7975239

[304] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: A survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015. [Online]. Available: https://doi.org/10.1007/s10618-014-0365-y

[305] M. Evangelou and N. M. Adams, "An anomaly detection framework for cyber-security data," *Computers & Security*, vol. 97, p. 101941, Oct. 2020. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2020.101941

[306] M. Ahmed, A. Naser Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016. [Online]. Available: https://doi.org/10.1016/j.jnca.2015.11.016

[307] P. Bindu, P. S. Thilagam, and D. Ahuja, "Discovering suspicious behavior in multilayer social networks," *Computers in Human Behavior*, vol. 73, pp. 568–582, Aug 2017. [Online]. Available: https://doi.org/10.1016%2Fj.chb.2017.04.001

[308] P. V. Bindu and P. S. Thilagam, "Mining social networks for anomalies: Methods and challenges," *Journal of Network and Computer Applications*, vol. 68, pp. 213–229, 2016. [Online]. Available: https://doi.org/10.1016/j.jnca.2016.02.021

[309] V. K. Giri and S. Sachdeva, "Anomaly Detection in Social Networks," in *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. IEEE, Jan 2019. [Online]. Available: https://doi.org/10.1109%2Fconfluence.2019.8776957

[310] R. Yu, H. Qiu, Z. Wen, C. Lin, and Y. Liu, "A Survey on Social Media Anomaly Detection," *ACM SIGKDD Explorations Newsletter*, vol. 18, no. 1, pp. 1–14, Aug 2016. [Online]. Available: https://doi.org/10.1145%2F2980765.2980767

[311] X. Xu, H. Liu, and M. Yao, "Recent Progress of Anomaly Detection," *Complexity*, vol. 2019, 2019. [Online]. Available: https://doi.org/10.1155/2019/2686378

[312] I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," *SN Computer Science*, vol. 2, no. 3, Mar. 2021. [Online]. Available: http://dx.doi.org/10.1007/s42979-021-00592-x

[313] K. Arshad, R. F. Ali, A. Muneer, I. A. Aziz, S. Naseer, N. S. Khan, and S. M. Taib, "Deep Reinforcement Learning for Anomaly Detection: A Systematic Review," *IEEE Access*, vol. 10, p. 124017124035, 2022. [Online]. Available: http://dx.doi.org/10.1109/access.2022.3224023

[314] U. Garg, M. Kaur, M. Kaushik, and N. Gupta, "Detection of DDoS Attacks using Semi-Supervised based Machine Learning Approaches," *International Conference on Computational Methods in Science & Technology (ICCMST)*, vol. 2, no. 2, pp. 112–117, 2022. [Online]. Available: https://doi.org/10.1109/iccmst54943.2021.00033

[315] T. Khaund, K. K. Bandeli, M. N. Hussain, A. Obadimu, S. Al-Khateeb, and N. Agarwal, "Analyzing social and communication network structures of social bots and humans," *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2018*, vol. 1, no. 1, pp. 794–797, 2018. [Online]. Available: https://doi.org/10.1109/ASONAM.2018.8508665

[316] X. Zhu and A. B. Goldberg, *Introduction to semi-supervised learning*. Springer Nature, 2022.

[317] N. Ding, H. X. Ma, H. Gao, Y. H. Ma, and G. Z. Tan, "Real-time anomaly detection based on long short-Term memory and Gaussian Mixture Model," *Computers and Electrical Engineering*, vol. 79, 2019. [Online]. Available: https://doi.org/10.1016/j.compeleceng.2019.106458

[318] S. Ding, Z. Zhu, and X. Zhang, "An overview on semi-supervised support vector machine," *Neural Computing and Applications*, vol. 28, no. 5, pp. 969–978, Nov 2015. [Online]. Available: https://doi.org/10.1007%2Fs00521-015-2113-7

[319] X. Sun, C. Zhang, G. Li, D. Sun, F. Ren, A. Zomaya, and R. Ranjan, "Detecting users' anomalous emotion using social media for business intelligence," *Journal of Computational Science*, vol. 25, pp. 193–200, 2018. [Online]. Available: http://dx.doi.org/10.1016/j.jocs.2017.05.029

[320] Y.-F. Li and Z.-H. Zhou, "Improving Semi-Supervised Support Vector Machines Through Unlabeled Instances Selection," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2011, pp. 386–391. [Online]. Available: https://doi.org/10.1609/aaai.v25i1.7920

[321] D. Pan, L. Nie, W. Kang, and Z. Song, "UAV Anomaly Detection Using Active Learning and Improved S3VM Model," in *2020 International Conference on Sensing, Measurement & Data Analytics in the era of Artificial Intelligence (ICSMD)*, 2020, pp. 253–258. [Online]. Available: https://doi.org/10.1109/ICSMD50554.2020.9261709

[322] Z. Yang, W. Cohen, and R. Salakhudinov, "Revisiting Semi-Supervised Learning with Graph Embeddings," in *Proceedings of The 33rd International Conference on Machine Learning*, 2016, pp. 40–48. [Online]. Available: https://proceedings.mlr.press/v48/yanga16.html

[323] S. M. Erfani, S. Rajasegarar, S. Karunasekera, and C. Leckie, "High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning," *Pattern Recognition*, vol. 58, pp. 121–134, 2016. [Online]. Available: https://doi.org/10.1016/j.patcog.2016.03.028

[324] M. F. Umer, M. Sher, and Y. Bi, "Flow-based intrusion detection: Techniques and challenges," *Computers & Security*, vol. 70, pp. 238–254, 2017. [Online]. Available: https://doi.org/10.1016/j.cose.2017.05.009

[325] NetFort Technologies Limited, "Flow Analysis Versus Packet Analysis . What Should You Choose ?" *White Paper*, p. 6, 2014.

[326] F. Mouton, L. Leenen, and H. Venter, "Social engineering attack examples, templates and scenarios," *Computers & Security*, vol. 59, p. 186209, Jun. 2016. [Online]. Available: http://dx.doi.org/10.1016/j.cose.2016.03.004

[327] L. Salvati, M. d'Amore, A. Fiorentino, A. Pellegrino, P. Sena, and F. Villecco, "Development and testing of a methodology for the assessment of acceptability of LKA systems," *Machines*, vol. 8, 2020. [Online]. Available: https://doi.org/10.3390/MACHINES8030047

[328] B. Camburn, V. Viswanathan, J. Linsey, D. Anderson, D. Jensen, R. Crawford, K. Otto, and K. Wood, "Design prototyping methods: State of the art in strategies, techniques, and guidelines," *Design Science*, vol. 3, no. Schrage 1993, pp. 1–33, 2017. [Online]. Available: https://doi.org/10.1017/dsj.2017.10

[329] E. C. D. Santos, P. Vilain, and D. H. Longo, "A systematic literature review to support the selection of user acceptance testing techniques," in *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*. ACM, May 2018. [Online]. Available: https://doi.org/10.1145%2F3183440.3195036

[330] D. Chicco and G. Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," *BMC Genomics*, vol. 21, no. 1, Jan 2020. [Online]. Available: https://doi.org/10.1186%2Fs12864-019-6413-7

[331] Q. Zhu, "On the performance of Matthews correlation coefficient (MCC) for imbalanced dataset," *Pattern Recognition Letters*, vol. 136, pp. 71–80, 2020. [Online]. Available: https://doi.org/10.1016/j.patrec.2020.03.030

[332] D. Nevado-Cataln, S. Pastrana, N. Vallina-Rodriguez, and J. Tapiador, "An analysis of fake social media engagement services," *Computers & Security*, vol. 124, 2023. [Online]. Available: https://doi.org/10.1016/j.cose.2022.103013

[333] Y. Jiang, J. Wang, Y. Liang, and J. Xia, "Combining static and dynamic features for real-time moving pedestrian detection," *Multimedia Tools and Applications*, vol. 78, pp. 3781–3795, 2019. [Online]. Available: https://doi.org/10.1007/s11042-018-6057-7

[334] A. Leontjeva and I. Kuzovkin, "Combining Static and Dynamic Features for Multivariate Sequence Classification," *IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 21–30, Dec 2017. [Online]. Available: https://doi.org/10.1109/DSAA.2016.10

[335] A. K. Shukla, "An Efficient Hybrid Evolutionary Approach for Identification of Zero-Day Attacks on Wired/Wireless Network System," *Wireless Personal Communications*, vol. 123, pp. 1–29, 2022. [Online]. Available: https://doi.org/10.1007/s11277-020-07808-y

[336] S. K. Uppada, K. Manasa, B. Vidhathri, R. Harini, and B. Sivaselvan, "Novel approaches to fake news and fake account detection in OSNs: user social engagement and visual content centric model," *Social Network Analysis and Mining*, vol. 12, 2022. [Online]. Available: https://doi.org/10.1007/s13278-022-00878-9

[337] Github, "Gaussian mixture model," https://github.com/guidesanti/machine-learning-course/tree/master/machine-learning-ex8/machine-learning-ex8/ex8, 2018, [Online; accessed 01-Aug-2022].

[338] GitHub, "Python implementations for semi-supervised learning," https://github.com/tmadl/semisup-learn, 2015, [Online; accessed 01-Jun-2022].

Appendix

Appendix A

Semi-supervised machine learning algorithms

This section provides the mathematical equations of the SSML algorithms used in this thesis. The Gaussian mixture model (GMM) algorithm is presented next.

A.1 Formal description of the GMM algorithm

Given a data set $\{x_j^1, x_j^2, \ldots, x_j^n\}$ of labelled and unlabelled data samples, one can model the mean (μ), variance (Σ) and probability ($\mathbb{P}$) as follows:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_j^i \tag{A.1}$$

$$\Sigma = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)(x_i - \mu)^T \tag{A.2}$$

Given a new unlabelled data point $x_{\text{unlabelled}}$, compute

$$\mathbb{P}(x_{\text{unlabelled}}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) \tag{A.3}$$

Predict

$$\text{Predict} = \begin{cases} \text{malicious} & \text{for} \quad \mathbb{P}(x_{\text{unlabelled}}) < \varepsilon \\ \text{normal} & \text{for} \quad \text{otherwise} \end{cases} \tag{A.4}$$

where ε is the threshold automatically determined during the training phase. The main advantage of GMM is that it captures the correlation of the input features naturally. Specifically, Σ is

an $n-$dimensional correlation matrix, However, computing Σ can be computationally expensive if n is large [301, 236]. The Python code for GMM is found in [301] and the Matlab version in [337]. The S3VM algorithm is algorithm is presented next.

A.2 Formal description of the S3VM algorithm

(i) Let (x_i, y_i) denote a sample data set, given a set of l labelled samples and u unlabelled samples. Further, let a hyperplane be denoted as $(w \cdot x) + b = 0$, then solve the following objective function over w, b, η, ξ and z.

$$\min_{w,b,\eta,\xi,z} \left[\|w\| + C\left(\sum_{i=1}^{l} \eta_i + \sum_{j=l+1}^{l+u} \min(\xi_j, z_j) \right) \right] \tag{A.5}$$

The first part of equation A.5 is the standard SVM where l slack variables η_i are imposed to ensure a soft margin for the labelled samples. The unlabelled samples can be labelled as either $+1$ or -1, thus, xi_j and z_j slack variables are imposed subjected to

$$\begin{cases} y_i(w^T \cdot x_i + b) + \eta_i \geq 1 \quad \text{and} \quad \eta_i \geq 0, \forall\ i = 1, \ldots, l \\ (w^T \cdot x_j - b) + \xi_j \geq 1 \quad \text{and} \quad \xi_j \geq 0, \forall\ j = l+1, \ldots, l+u \\ -(w^T \cdot x_j - b) + z_j \geq 1 \quad \text{and} \quad z_j \geq 0, \forall\ j = l+1, \ldots, l+u \end{cases} \tag{A.6}$$

where $C \in (0, 1]$ is a fixed misclassification penalty parameter. In equation A.6, the first constraint is limited to the standard SVM. The second and third constraints consider the possibility that an unlabelled sample could be labelled as $+1$ or -1. To summarise equations A.5 and A.6 in a less technical manner,

1. Begin with a small set of labelled data and a large set of unlabelled data points;
2. Train the standard SVM on the labelled data points and find the initial decision boundary;
3. Label the unlabelled data points using the current standard SVM;
4. Incorporate the predicted labels, re-train the SVM model and find a new decision boundary;
5. Iteratively label the unlabelled data points and update the decision boundary until convergence is reached.

The Python code for S3VM is found in [338, 236]. The LP algorithm is algorithm is presented next.

A.3 Formal description of the LP algorithm

Given a data set with N labelled and M unlabelled data points, i.e.,:

$$\begin{cases} X = (X_L, X_U) \text{ where } X \in \mathbb{R} \\ Y = (Y_L, Y_U) \text{ where } Y \in \{-1, +1, 0\} \end{cases} \tag{A.7}$$

Using the k-nearest neighbours (kNN) model, compute the affinity matrix W and degree matrix D

$$D = diag\left(\left|\sum_j W_{ij}\right| \forall i = 1, 2, \ldots, N+M\right) \tag{A.8}$$

Let $Y^{(0)} = Y$ and $Y_L = \{y_0, y_1, \ldots, y_N\}$. Then iterate until convergence is achieved.

$$\begin{cases} Y^{(t+1)} = D^{-1} W Y^{(t)} \\ Y_L^{(t+1)} = Y_L \end{cases} \tag{A.9}$$

Proof of convergence and Python code is found in [236]. The LS algorithm is algorithm is presented next.

A.4 Formal description of the LS algorithm

Using the k-nearest neighbours (kNN) model, compute the affinity matrix W and degree matrix D

$$D = diag\left(\left|\sum_j W_{ij}\right| \forall i = 1, 2, \ldots, N+M\right) \tag{A.10}$$

Compute the normalised graph Laplacian:

$$L = D^{-\frac{1}{2}} W D^{-\frac{1}{2}} \tag{A.11}$$

Choose $\alpha \in (0, 1]$, and iterate until convergence is achieved

$$Y^{(t+1)} = \alpha L Y^{(t)} + (1 - \alpha) Y^{(0)} \tag{A.12}$$

Proof of convergence and Python code is found in [236]. The one-class support vector machines (OCSVM) algorithm is presented next.

A.5 Formal description of the OCSVM

(i) Given a training data set $Z = \{z_1, z_2, \ldots, z_N\}$ of one class and let $\phi : Z \longrightarrow G$ be a kernel map that maps training data points to another space G. Then, to separate data sets from the origin, the following quadratic programming equation must be solved:

$$\min_{w \in G, \xi_i, b \in \mathbb{R}} \left\{ \frac{1}{2}\|w\|^2 + \frac{1}{\nu N} \sum_{i=1}^{N} (\xi_i - b) \right\} \tag{A.13}$$

subject to

$$\nu \in (0, 1],\ \xi_i \geq 0,\ \forall_i = 1, 2, \ldots, N \text{ and } (w \cdot \phi(z_i) \geq b - \xi_i,\ \forall_i = 1, 2, \ldots, N.) \tag{A.14}$$

where ξ_i is a non-zero slack and ν is the ratio of anomalies in the training data set, which is set to $\nu = 0.1$ in this thesis. Finally, the decision boundary function $f(z)$ becomes

$$f(z) = sign\{(w \cdot \phi(z)) - b\}.$$

Appendix B

Mann-Whitney U test

This appendix contains MannWhitney U test results on combining 𝕏 data sets to prove that no bias was introduced by combining the data sets. The MannWhitney U test goodness-of-fit is formulated as follows:

$$H_0 = \text{no difference between two distributions.}$$

$$H_1 = \text{difference between two distributions.}$$

If p-value < 0.05, H_0 is rejected; else, H_0 cannot be rejected. The results are summarised below.

Feature	**[5] vs combined**	**[6] vs combined**	**[7] vs combined**
Favourites_count	H_0 cannot be rejected	H_0 cannot be rejected	H_0 cannot be rejected
Lists_count	H_0 cannot be rejected	H_0 cannot be rejected	H_0 cannot be rejected
Statuses_count	H_0 cannot be rejected	H_0 cannot be rejected	H_0 cannot be rejected
Status.retweet_count	H_0 cannot be rejected	H_0 cannot be rejected	H_0 cannot be rejected
Friends_count	H_0 cannot be rejected	H_0 cannot be rejected	H_0 cannot be rejected
Followers_count	H_0 cannot be rejected	H_0 cannot be rejected	H_0 cannot be rejected

Table B.1: Mann-Whitney U test goodness-of-fit results

Based on the results in Table B.1 it is clear that no bias was introduced on these features by combining the data sets.

Appendix C

Data set features

This appendix contains features of the network intrusion data sets used in this thesis, namely IoT Intrusion-2020, UNSW-NB15, CICDDOS2019, and CIRACICDOHBRW2020.

C.1 The IoT Intrusion-2020 data set

This section contains the IoT Intrusion-2020 features.

No.	Feature name	Data type	Min	Max
1	“Flow_ID”	Non-numeric	N/A	N/A
2	“Source_IP”	Non-numeric	N/A	N/A
3	“Source_Port”	Numeric	0	65500
4	“Destination_IP”	Non-numeric	N/A	N/A
5	“Destination_Port”	Numeric	0	65371
6	“Protocol”	Numeric	0	17
7	“Timestamp”	Non-numeric	N/A	N/A
8	“Flow_Duration”	Numeric	0	99984
9	“Total_Forward_Packets”	Numeric	0	186
10	“Total_Backward_Packets”	Numeric	1	560
11	“TotalLength_Forward_Packets”	Numeric	0	109846
12	“TotalLength_Backward_Packets”	Numeric	0	773284
13	“Forward_Packet_Length_Max”	Numeric	0	1464
14	“Forward_Packet_Length_Min”	Numeric	0	1464
15	“Forward_Packet_Length_Mean”	Numeric	0	1464
16	“Forward_Packet_Length_Std”	Numeric	0	1032.375901
17	“Backward_Packet_Length_Max”	Numeric	0	1464
18	“Backward_Packet_Length_Min”	Numeric	0	1460
19	“Backward_Packet_Length_Mean”	Numeric	0	1460
20	“Backward_Packet_Length_Std”	Numeric	0	1032.375901
21	“Flow_bytes/s”	Numeric	0	4066000000
22	“Flow_Packet/s”	Numeric	20.00540146	5000000
23	“Flow_IAT_Mean”	Numeric	0	99973
24	“Flow_IAT_Std”	Numeric	0	67901.34288
25	“Flow_IAT_Max”	Numeric	0	99973
26	“Flow_IAT_Min”	Numeric	0	99973

Table C.1: Features of the IoT Intrusion-2020 data set adopted from [8]

No.	Feature name	Data type	Min	Max
27	“Forward_IAT_Total”	Numeric	0	99676
28	“Forward_IAT_Mean”	Numeric	0	98135
29	“Forward_IAT_Std”	Numeric	0	70374.09529
30	“Forward_IAT_Max”	Numeric	0	99600
31	“Forward_IAT_Min”	Numeric	0	98135
32	“Backward_IAT_Total”	Numeric	0	99973
33	“Backward_IAT_Mean”	Numeric	0	99973
34	“Backward_IAT_Std”	Numeric	0	68998.06549
35	“Backward_IAT_Max”	Numeric	0	99973
36	“Backward_IAT_Min”	Numeric	0	99973
37	“Forward_PSH_Flags”	Numeric	0	0
38	“Backward_PSH_Flags”	Numeric	0	1
39	“Forward_URG_Flags”	Numeric	0	0
40	“Backward_URG_Flags”	Numeric	0	1
41	“Forward_Header_Length”	Numeric	0	3832
42	“Backward_Header_Length”	Numeric	0	17920
43	“Forward_Packet/s”	Numeric	0	4000000
44	“Backward_Packet/s”	Numeric	0	1000000
45	“Packet_Length_Min”	Numeric	0	1460
46	“Packet_Length_Max”	Numeric	0	1464
47	“Packet_Length_Mean”	Numeric	0	1460
48	“Packet_Length_Std”	Numeric	0	842.931393
49	“Packet_Length_Var”	Numeric	0	710533.3333
50	“FIN_Flag_Count”	Numeric	0	1
51	“SYN_Flag_Count”	Numeric	0	1
52	“RST_Flag_Count”	Numeric	0	1
53	“PSH_Flag_Count”	Numeric	0	1
54	“ACK_Flag_Count”	Numeric	0	1
55	“URG_Flag_Count”	Numeric	0	1
56	“CWE_Flag_Count”	Numeric	0	1
57	“ECE_Flag_Count”	Numeric	0	1

Table C.2: *(Continued.)* Features of the IoT Intrusion-2020 data set adopted from [8]

No.	Feature name	Data type	Min	Max
58	“Down/Up_Ratio”	Numeric	0	14
59	“Packet_Size_Avg”	Numeric	0	2190
60	“Forward_Segment_Size_Avg”	Numeric	0	1464
61	“Backward_Segment_Size_Avg”	Numeric	0	1460
62	“Forward_Bytes/b_Avg”	Numeric	0	0
63	“Forward_Packet/b_Avg”	Numeric	0	0
64	“Forward_Bulk_Rate_Avg”	Numeric	0	0
65	“Backward_Bytes/b_Avg”	Numeric	0	0
66	“Backward_Packet/b_Avg”	Numeric	0	0
67	“Backward_Bulk_Rate_Avg”	Numeric	0	0
68	“Subflow_Forward_Packet”	Numeric	0	186
69	“Subflow_Forward_Bytes”	Numeric	0	109846
70	“Subflow_Backward_Packet”	Numeric	1	560
71	“Subflow_Backward_Bytes”	Numeric	0	773284
72	“Init_Forward_Win_Bytes”	Numeric	-1	-1
73	“Init_Backward_Win_Bytes”	Numeric	-1	65535
74	“Forward_Act_Data_Packet”	Numeric	0	186
75	“Forward_Segment_Size_Min”	Numeric	0	0
76	“Active_Mean”	Numeric	0	9044.625
77	“Active_Std”	Numeric	0	8598.65825
78	“Active_Max”	Numeric	0	26785
79	“Active_Min”	Numeric	0	6659
80	“Idle_Mean”	Numeric	0	99973
81	“Idle_Std”	Numeric	0	67071.90662
82	“Idle_Max”	Numeric	0	99973
83	“Idle_Min”	Numeric	0	99973
84	“Label”	Non-numeric	N/A	N/A
85	“Category”	Non-numeric	N/A	N/A
86	“Sub_Category”	Non-numeric	N/A	N/A

Table C.3: *(Continued.)* Features of the IoT Intrusion-2020 data set adopted from [8]

The features of the IoT Intrusion-2020 data set indicated in Tables C.1, C.2 and C.3 consist of forward (i.e., source-to-destination) and backward (i.e., destination-to-source) network traffic flows. Furthermore, the majority of these features are numeric in nature, which means feature selection methods can easily be implemented [238, 244].

The features of the IoT Intrusion-2020 data set are listed below.

- "Flow_ID : Flow address."
- "Source_IP : Source IP address."
- "Source_Port : Source port address."
- "Destination_IP : Destination IP address."
- "Destination_Port : Destination port address."
- "Protocol : Network Protocol."
- "Timestamp: Simulation time stamp."
- "Flow_Duration : Duration of the flow in microsecond."
- "Total_Forward_Packets : Total packets in the forward direction."
- "Total_Backward_Packets : Total packets in the backward direction."
- "TotalLength_Forward_Packets : Total size of packet in forward direction."
- "TotalLength_Backward_Packets : Total size of packet in backward direction."
- "Forward_Packet_Length_Min : Minimum size of packet in forward direction."
- "Forward_Packet_Length_Max : Maximum size of packet in forward direction."
- "Forward_Packet_Length_Mean : Mean size of packet in forward direction."
- "Forward_Packet_Length_Std : Standard deviation size of packet in forward direction."
- "Backward_Packet_Length_Min : Minimum size of packet in backward direction."
- "Backward_Packet_Length_Max : Maximum size of packet in backward direction."
- "Backward_Packet_Length_Mean : Mean size of packet in backward direction."

- "Backward_Packet_Length_Std : Standard deviation size of packet in backward direction."
- "Flow_bytes/s : Number of flow bytes per second."
- "Flow_Packet/s: Number of flow packets per second."
- "Flow_IAT_Mean : Mean time between two packets sent in the flow."
- "Flow_IAT_Std : Standard deviation time between two packets sent in the flow."
- "Flow_IAT_Max : Maximum time between two packets sent in the flow."
- "Flow_IAT_Min : Minimum time between two packets sent in the flow."
- "Forward_IAT_Min : Minimum time between two packets sent in the forward direction."
- "Forward_IAT_Max : Maximum time between two packets sent in the forward direction."
- "Forward_IAT_Mean : Mean time between two packets sent in the forward direction."
- "Forward_IAT_Std : Standard deviation time between two packets sent in the forward direction."
- "Forward_IAT_Total : Total time between two packets sent in the forward direction."
- "Backward_IAT_Min : Minimum time between two packets sent in the backward direction."
- "Backward_IAT_Max : Maximum time between two packets sent in the backward direction."
- "Backward_IAT_Mean : Mean time between two packets sent in the backward direction."
- "Backward_IAT_Std : Standard deviation time between two packets sent in the backward direction."
- "Backward_IAT_Total : Total time between two packets sent in the backward direction."
- "Forward_PSH_Flags : Number of times the PSH flag was set in packets travelling in the forward direction."
- "Backward_PSH_Flags : Number of times the PSH flag was set in packets travelling in the backward direction."

- "Forward_URG_Flags : Number of times the URG flag was set in packets travelling in the forward direction."
- "Backward_URG_Flags : Number of times the URG flag was set in packets travelling in the backward direction."
- "Forward_Header_Length : Total bytes used for headers in the forward direction."
- "Backward_Header_Length : Total bytes used for headers in the backward direction."
- "Forward_Packet/s : Number of forward packets per second."
- "Backward_Packet/s : Number of backward packets per second."
- "Packet_Length_Min : Minimum length of a packet."
- "Packet_Length_Max : Maximum length of a packet."
- "Packet_Length_Mean : Mean length of a packet."
- "Packet_Length_Std : Standard deviation length of a packet."
- "Packet_Length_Var : Variance length of a packet."
- "FIN_Flag_Count : Number of packets with FIN."
- "SYN_Flag_Count : Number of packets with SYN."
- "RST_Flag_Count: Number of packets with RST."
- "PSH_Flag_Count : Number of packets with PSH."
- "ACK_Flag_Count : Number of packets with ACK."
- "URG_Flag_Count : Number of packets with URG."
- "CWE_Flag_Count : Number of packets with CWE."
- "ECE_Flag_Count : Number of packets with ECE."
- "Down/Up_Ratio : Download and upload ratio."
- "Packet_Size_Avg : Average size of packet."
- "Forward_Segment_Size_Avg : Average size observed in the forward direction."

- "Backward_Segment_Size_Avg: Average size observed in the backward direction."
- "Forward_Bytes/b_Avg : Average number of bytes bulk rate in the forward direction."
- "Forward_Packet/b_Avg : Average number of packets bulk rate in the forward direction."
- "Forward_Bulk_Rate_Avg : Average number of bulk rate in the forward direction."
- "Backward_Bytes/b_Avg: Average number of bytes bulk rate in the backward direction."
- "Backward_Packet/b_Avg : Average number of packets bulk rate in the backward direction."
- "Backward_Bulk_Rate_Avg : Average number of bulk rate in the backward direction."
- "Sub-flow_Forward_Packet : The average number of packets in a sub-flow in the forward direction."
- "Sub-flow_Forward_Bytes : The average number of bytes in a sub-flow in the forward direction."
- "Sub-flow_Backward_Packet : The average number of packets in a sub-flow in the backward direction."
- "Sub-flow_Backward_Bytes : The average number of bytes in a sub-flow in the backward direction."
- "Init_Forward_Win_Bytes : The total number of bytes sent in initial window in the forward direction."
- "Init_Backward_Win_Bytes : The total number of bytes sent in initial window in the backward direction."
- "Forward_Act_Data_Packet : Count of packets with at least 1 byte of TCP data payload in the forward direction."
- "Forward_Segment_Size_Min : Minimum segment size observed in the forward direction."
- "Active_Min : Minimum time a flow was active before becoming idle."
- "Active_Mean : Mean time a flow was active before becoming idle."

- "Active_Max: Maximum time a flow was active before becoming idle."
- "Active_Std : Standard deviation time a flow was active before becoming idle."
- "Idle_Min : Minimum time a flow was idle before becoming active."
- "Idle_Mean : Mean time a flow was idle before becoming active."
- "Idle_Max : Maximum time a flow was idle before becoming active."
- "Idle_Std : Standard deviation time a flow was idle before becoming active."
- Label: Attack label i.e. normal or anomaly network traffic.
- Category: Attack main category in Table 5.1.
- Sub-Category: Attack sub-category in Table 5.1.

C.2 UNSW-NB15 features

This section contains the UNSW-NB15 features.

No.	Feature name	Data type	Min	Max
1	“dur”	Numeric	0	59.999989
2	“proto”	Non-numeric	N/A	N/A
3	“service”	Non-numeric	N/A	N/A
4	“state”	Non-numeric	N/A	N/A
5	“spkts”	Numeric	1	9616
6	“dpkts”	Numeric	0	10974
7	“sbytes”	Numeric	28	12965233
8	“dbytes”	Numeric	0	14655550
9	“rate”	Numeric	0	1000000.003
10	“sttl”	Numeric	0	255
11	“dttl”	Numeric	0	254
12	“sload”	Numeric	0	5988000256
13	“dload”	Numeric	0	22422730
14	“sloss”	Numeric	0	4803
15	“dloss”	Numeric	0	5484
16	“sinpkt”	Numeric	0	84371.496
17	“dinpkt”	Numeric	0	56716.824
18	“sjit”	Numeric	0	1460480.016
19	“djit”	Numeric	0	289388.2697
20	“swin”	Numeric	0	255
21	“stcpb”	Numeric	0	4294958913
22	“dtcpb”	Numeric	0	4294881924
23	“dwin”	Numeric	0	255
24	“tcprtt”	Numeric	0	2.518893
25	“synack”	Numeric	0	2.100352

Table C.4: Features of the UNSW-NB15 data set

No.	Feature name	Data type	Min	Max
26	“ackdat”	Numeric	0	1.520884
27	“smean”	Numeric	24	1504
28	“dmean”	Numeric	0	1458
29	“trans_depth”	Numeric	0	172
30	“response_body_len”	Numeric	0	6558056
31	“ct_srv_src”	Numeric	1	63
32	“ct_state_ttl”	Numeric	0	6
33	“ct_dst_ltm”	Numeric	1	51
34	“ct_src_dport_ltm”	Numeric	1	51
35	“ct_dst_sport_ltm”	Numeric	1	46
36	“ct_dst_src_ltm”	Numeric	1	65
37	“is_ftp_login”	Numeric	0	4
38	“ct_ftp_cmd”	Numeric	0	4
39	“ct_flw_http_mthd”	Numeric	0	30
40	“ct_src_ltm”	Numeric	1	60
41	“ct_srv_dst”	Numeric	1	62
42	“is_sm_ips_ports”	Numeric	0	1
43	“attack_cat”	Non-numeric	N/A	N/A
44	“label”	Non-numeric	N/A	N/A

Table C.5: *(Continued.)* Features of the UNSW-NB15 data set

Next, features of the UNSW-NB15 data set are described.

- “dur : Record total duration.”
- “proto : Transaction protocol.”
- “service : Source port address.”
- “state : Contains the state and its dependent protocol.”
- “spkts : Source to destination packet count.”
- “dpkts : Destination to source packet count.”

- “sbytes: Source to destination transaction bytes.”
- “dbytes : Destination to source transaction bytes.”
- “rate : Ethernet data rates transmitted and received.”
- “sttl : Source to destination time to live value.”
- “dttl : Destination to source time to live value.”
- “sload : Source bits per second.”
- “dload : Destination bits per second.”
- “sloss : Source packets retransmitted or dropped.”
- “dloss : Destination packets retransmitted or dropped.”
- “sinpkt : Source interpacket arrival time (mSec).”
- “dinpkt : Destination interpacket arrival time (mSec).”
- “sjit : Source jitter (mSec).”
- “djit : Destination jitter (mSec).”
- “swin : Source TCP window advertisement value.”
- “stcpb : Source TCP base sequence number.”
- “dtcpb: Destination TCP base sequence number.”
- “dwin : Destination TCP window advertisement value.”
- “tcprtt : TCP connection setup round-trip time, the sum of ’synack’ and ’ackdat’.”
- “synack : TCP connection setup time, the time between the SYN and the SYN_ACK packets.”
- “ackdat : TCP connection setup time, the time between the SYN_ACK and the ACK packets.”
- “smean : Mean of the packet size transmitted by the src.”
- “dmean : Mean of the packet size transmitted by the dst.”

- "trans_depth : Represents the pipelined depth into the connection of http request or response transaction."
- "response_body_len : Actual uncompressed content size of the data transferred from the server's http service."
- "ct_srv_src : No. of connections that contain the same service (14) and source address (1) in 100 connections according to the last time (26)."
- "ct_state_ttl : No. for each state (6) according to specific range of values for source or destination time to live (10) (11)."
- "ct_dst_ltm :No. of connections of the same destination address (3) in 100 connections according to the last time (26)."
- "ct_src_dport_ltm : No. of connections of the same source address (1) and the destination port (4) in 100 connections according to the last time (26)."
- "ct_dst_sport_ltm : No. of connections of the same destination address (3) and the source port (2) in 100 connections according to the last time (26)."
- "ct_dst_src_ltm : No. of connections of the same source (1) and the destination (3) address in in 100 connections according to the last time (26)."
- "is_ftp_login : If the ftp session is accessed by user and password then 1 else 0."
- "ct_ftp_cmd : No of flows that has a command in ftp session."
- "ct_flw_http_mthd : No. of flows that has methods such as Get and Post in http service."
- "ct_src_ltm : No. of connections of the same source address (1) in 100 connections according to the last time (26)."
- "ct_srv_dst : No. of connections that contain the same service (14) and destination address (3) in 100 connections according to the last time (26)."
- "is_sm_ips_ports : If source (1) and destination (3)IP addresses equal and port numbers (2)(4) equal then, this variable takes value 1 else 0."
- "attack_cat : The name of each attack category. In this data set , nine categories e.g. Fuzzers, Analysis, Backdoors, DoS Exploits, Generic, Reconnaissance, Shellcode and Worms."

- "label : 0 for normal and 1 for attack records."

C.3 CICDDOS2019 features

This section contains CICDDOS2019 features.

No.	Feature name	Data type	Min	Max
1	"Flow_ID"	Non-numeric	N/A	N/A
2	"Source_IP"	Non-numeric	N/A	N/A
3	"Source_Port"	Numeric	0	65532
4	"Destination_IP"	Non-numeric	N/A	N/A
5	"Destination_Port"	Numeric	0	65535
6	"Protocol"	Numeric	0	17
7	"Timestamp"	Non-numeric	N/A	N/A
8	"Flow_Duration"	Numeric	0	119999992
9	"Total_Forward_Packets"	Numeric	1	8456
10	"Total_Backward_Packets"	Numeric	0	421
11	"TotalLength_Forward_Packets"	Numeric	0	67750
12	"TotalLength_Backward_Packets"	Numeric	0	670016
13	"Forward_Packet_Length_Max"	Numeric	0	3087
14	"Forward_Packet_Length_Min"	Numeric	0	1715
15	"Forward_Packet_Length_Mean"	Numeric	0	2595.555556
16	"Forward_Packet_Length_Std"	Numeric	0	1168.847509
17	"Backward_Packet_Length_Max"	Numeric	0	3500
18	"Backward_Packet_Length_Min"	Numeric	0	520
19	"Backward_Packet_Length_Mean"	Numeric	0	1946.666667
20	"Backward_Packet_Length_Std"	Numeric	0	1469.59178

Table C.6: Features of the CICDDOS2019 data set

No.	Feature name	Data type	Min	Max
21	“Flow_bytes/s”	Numeric	0	2944000000
22	“Flow_Packet/s”	Numeric	1	3000000
23	“Flow_IAT_Mean”	Numeric	0	15914398.67
24	“Flow_IAT_Std”	Numeric	0	35961975.76
25	“Flow_IAT_Max”	Numeric	0	95146794
26	“Flow_IAT_Min”	Numeric	0	6369775
27	“Forward_IAT_Total”	Numeric	0	119999992
28	“Forward_IAT_Mean”	Numeric	0	23962226.67
29	“Forward_IAT_Std”	Numeric	0	41503790.59
30	“Forward_IAT_Max”	Numeric	0	95146794
31	“Forward_IAT_Min”	Numeric	0	6369775
32	“Backward_IAT_Total”	Numeric	0	119999989
33	“Backward_IAT_Mean”	Numeric	0	23962148.33
34	“Backward_IAT_Std”	Numeric	0	41503653.18
35	“Backward_IAT_Max”	Numeric	0	71886439
36	“Backward_IAT_Min”	Numeric	0	187
37	“Forward_PSH_Flags”	Numeric	0	1
38	“Backward_PSH_Flags”	Numeric	0	0
39	“Forward_URG_Flags”	Numeric	0	0
40	“Backward_URG_Flags”	Numeric	0	0
41	“Forward_Header_Length”	Numeric	0	7192
42	“Backward_Header_Length”	Numeric	0	8444
43	“Forward_Packet/s”	Numeric	0	3000000
44	“Backward_Packet/s”	Numeric	0	2000000
45	“Packet_Length_Min”	Numeric	0	1472
46	“Packet_Length_Max”	Numeric	0	3500
47	“Packet_Length_Mean”	Numeric	0	2294.285714
48	“Packet_Length_Std”	Numeric	0	1416.666873
49	“Packet_Length_Var”	Numeric	0	2006945.029
50	“FIN_Flag_Count”	Numeric	0	0

Table C.7: *(Continued.)* Features of the CICDDOS2019 data set

No.	Feature name	Data type	Min	Max
51	“SYN_Flag_Count”	Numeric	0	1
52	“RST_Flag_Count”	Numeric	0	1
53	“PSH_Flag_Count”	Numeric	0	0
54	“ACK_Flag_Count”	Numeric	0	1
55	“URG_Flag_Count”	Numeric	0	1
56	“CWE_Flag_Count”	Numeric	0	1
57	“ECE_Flag_Count”	Numeric	0	0
58	“Down/Up_Ratio”	Numeric	0	12
59	“Packet_Size_Avg”	Numeric	0	2409
60	“Forward_Segment_Size_Avg”	Numeric	0	2595.555556
61	“Backward_Segment_Size_Avg”	Numeric	0	1946.666667
62	“Forward Header Length”	Numeric	0	7192
63	“Forward Avg Bytes/Bulk”	Numeric	0	0
64	“Forward Avg Packets/Bulk”	Numeric	0	0
65	“Forward Avg Packets/Bulk”	Numeric	0	0
66	“Backward Avg Bytes/Bulk”	Numeric	0	0
67	“Backward Avg Packets/Bulk”	Numeric	0	0
68	“Backward Avg Bulk Rate”	Numeric	0	0
69	“Subflow_Forward_Packet”	Numeric	1	8456
70	“Subflow_Forward_Bytes”	Numeric	0	67750
71	“Subflow_Backward_Packet”	Numeric	0	421
72	“Subflow_Backward_Bytes”	Numeric	0	670016
73	“Init_Forward_Win_Bytes”	Numeric	-1	65535
74	“Init_Backward_Win_Bytes”	Numeric	-1	65535
75	“Forward_Act_Data_Packet”	Numeric	0	94
76	“Forward_Segment_Size_Min”	Numeric	0	1472
77	“Active_Mean”	Numeric	0	18300116
78	“Active_Std”	Numeric	0	22840639.74
79	“Active_Max”	Numeric	0	53761691
80	“Active_Min”	Numeric	0	14619917

Table C.8: *(Continued.)* Features of the CICDDOS2019 data set

No.	Feature name	Data type	Min	Max
81	“Idle_Mean”	Numeric	0	95146794
82	“Idle_Std”	Numeric	0	33338860.02
83	“Idle_Max”	Numeric	0	95146794
84	“Idle_Min”	Numeric	0	95146794
85	“Similar HTTP”	Non-numeric	N/A	N/A
86	“Inbound”	Numeric	0	1
87	“Label”	Non-numeric	N/A	N/A

Table C.9: *(Continued.)* Highlights features of the CICDDOS2019 data set

C.4 CIRACICDoHBrw2020 features

This section contains the CIRACICDoHBrw2020 features.

No.	Feature name	Data type	Min	Max
1	“SourceIP”	Non-numeric	N/A	N/A
2	“DestinationIP”	Non-numeric	N/A	N/A
3	“SourcePort”	Numeric	443	65532
4	“DestinationPort”	Numeric	443	65497
5	“TimeStamp”	Non-numeric	N/A	N/A
6	“Duration”	Numeric	0	179.021144
7	“FlowBytesSent”	Numeric	55	8015359
8	“FlowSentRate”	Numeric	-1	23043478.26
9	“FlowBytesReceived”	Numeric	0	7723184
10	“FlowReceivedRate”	Numeric	0	7600000
11	“PacketLengthVariance”	Numeric	0	1578115.36
12	“PacketLengthStandardDeviation”	Numeric	0	1256.230616
13	“PacketLengthMean”	Numeric	55	689.8
14	“PacketLengthMedian”	Numeric	54	317
15	“PacketLengthMode”	Numeric	54	553

Table C.10: Features of the CIRACICDoHBrw2020 data set

No.	Feature name	Data type	Min	Max
16	“PacketLengthSkewFromMedian”	Numeric	-10	2.932374601
17	“PacketLengthSkewFromMode”	Numeric	-10	1.42617659
18	“PacketLengthCoefficientofVariation”	Numeric	0	3.100069681
19	“PacketTimeVariance”	Numeric	0	3546.090995
20	“PacketTimeStandardDeviation”	Numeric	0	59.54906376
21	“PacketTimeMean”	Numeric	0	113.1986567
22	“PacketTimeMedian”	Numeric	0	119.286385
23	“PacketTimeMode”	Numeric	0	119.990979
24	“PacketTimeSkewFromMedian”	Numeric	-10	2.846948507
25	“PacketTimeSkewFromMode”	Numeric	-10	12.95640605
26	“PacketTimeCoefficientofVariation”	Numeric	-1	5.616085483
27	“ResponseTimeVariance”	Numeric	-1	647.2453301
28	“ResponseTimeStandardDeviation”	Numeric	-1	25.44101669
29	“ResponseTimeMean”	Numeric	-1	28.017596
30	“ResponseTimeMedian”	Numeric	0	28.017596
31	“ResponseTimeMode”	Numeric	-1	28.017596
32	“ResponseTimeSkewFromMedian”	Numeric	-10	2.970715578
33	“ResponseTimeSkewFromMode”	Numeric	-10	5.428780989
34	“ResponseTimeCoefficientofVariation”	Numeric	0	66.30974712
35	Label	Non-numeric	N/A	N/A

Table C.11: *(Continued.)* Features of the CIRACICDoHBrw2020 data set

Next, features for the CIRACICDoHBrw2020 data set are described.

- “SourceIP: source IP address.”
- “DestinationIP: destination IP address.”
- “SourcePort: source port.”
- “DestinationPort: destination port.”
- “TimeStamp: time.”

- “Duration: record duration.”
- “FlowBytesSent: Number of flow bytes sent.”
- “FlowSentRate: Rate of flow bytes sent.”
- “FlowBytesReceived: Number of flow bytes received.”
- “FlowReceivedRate: Rate of flow bytes received.”
- “PacketLengthVariance: Variance of Packet Length.”
- “PacketLengthStandardDeviation: Standard Deviation of Packet Length.”
- “PacketLengthMean: Mean Packet Length.”
- “PacketLengthMedian: Median Packet Length”
- “PacketLengthMode: Mode Packet Length.”
- “PacketLengthSkewFromMedian: Skew from median Packet Length.”
- “PacketLengthSkewFromMode: Skew from mode Packet Length.”
- “PacketLengthCoefficientofVariation: Coefficient of Variation of Packet Length”
- “PacketTimeVariance: Variance of Packet Time.”
- “PacketTimeStandardDeviation: Standard Deviation of Packet Time.”
- “PacketTimeMean: Mean Packet Time.”
- “PacketTimeMedian: Median Packet Time.”
- “PacketTimeMode: Mode Packet Time.”
- “PacketTimeSkewFromMedian: Skew from median Packet Time.”
- “PacketTimeSkewFromMode: Skew from mode Packet Time.”
- “PacketTimeCoefficientofVariation: Coefficient of Variation of Packet Time.”
- “ResponseTimeVariance: Variance time response.”
- “ResponseTimeStandardDeviation: Standard deviation time response.”
- “ResponseTimeMean: Mean Request/response time difference.”

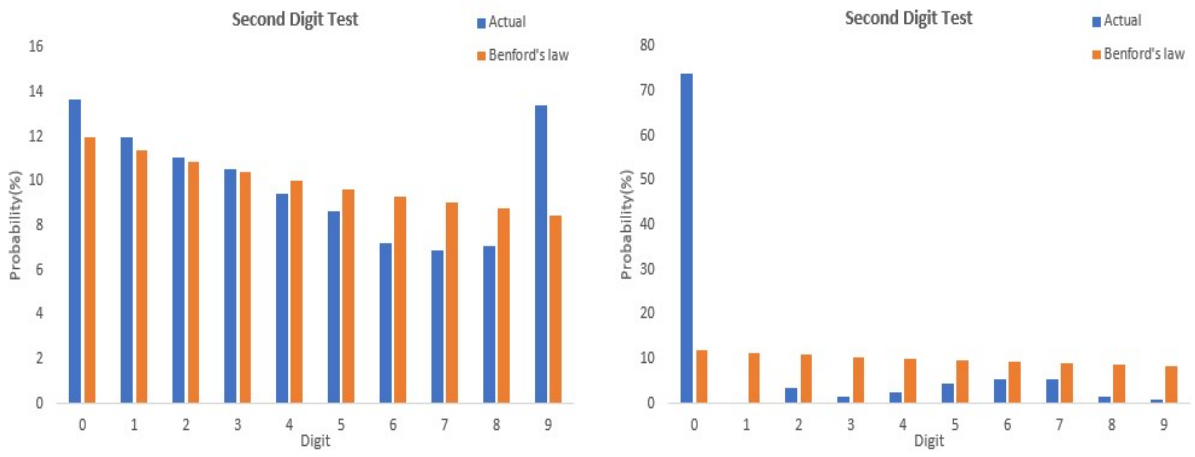
- “ResponseTimeMedian: Median Request/response time difference.”
- “ResponseTimeMode: Mode Request/response time difference.”
- “ResponseTimeSkewFromMedian: Skew from median Request/response time difference.”
- “ResponseTimeSkewFromMode: Skew from mode Request/response time difference.”
- “ResponseTimeCoefficientofVariation: Coefficient of Variation of Request/response time difference.”

Appendix D

Feature selection results on NIDS data sets

This chapter provides feature selection results on NIDS data sets.

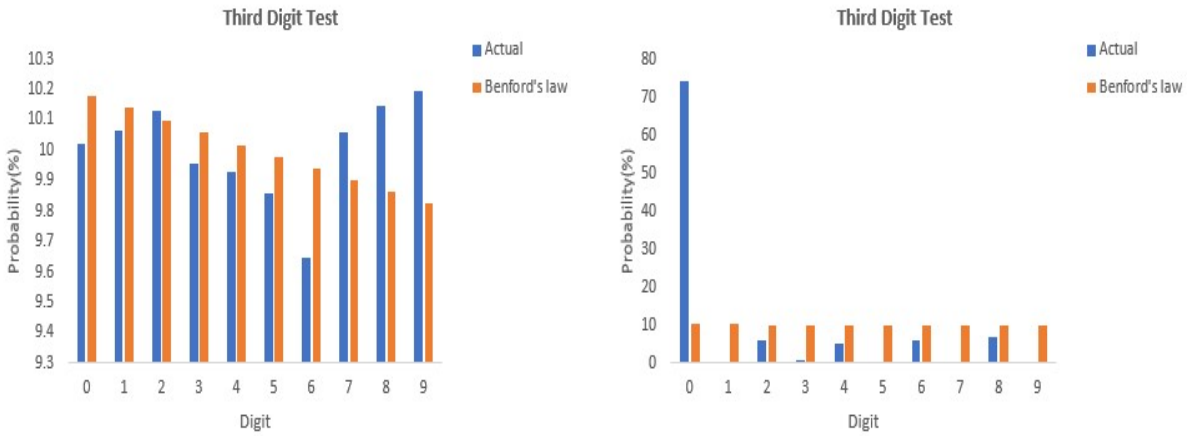
D.1 UNSW-NB15 Feature selection results



(a) Benign network traffic.

(b) Malicious zero-day network traffic.

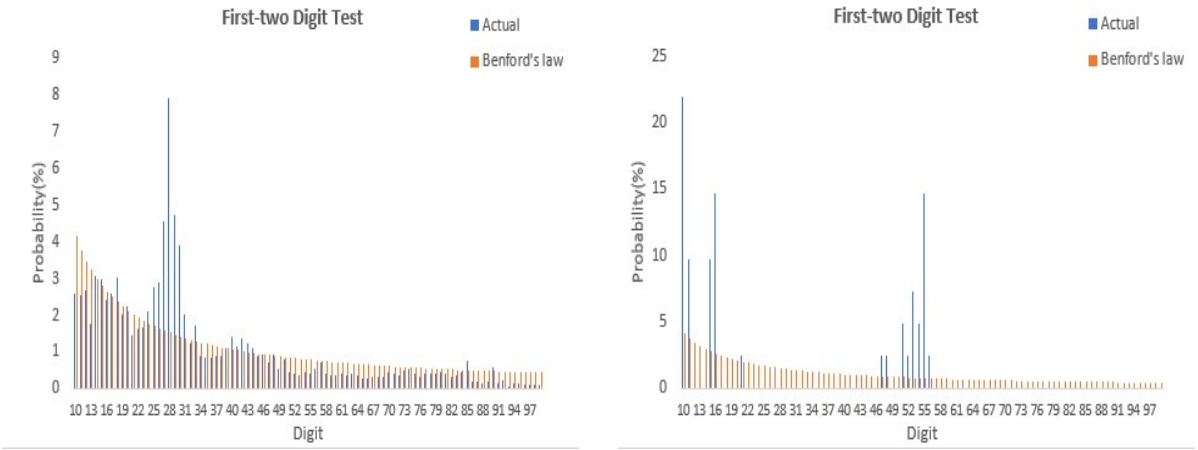
Figure D.1: Comparing SDT benign and zero-day network traffic on the UNSW-NB15 data set



(a) Benign network traffic.

(b) Malicious zero-day network traffic.

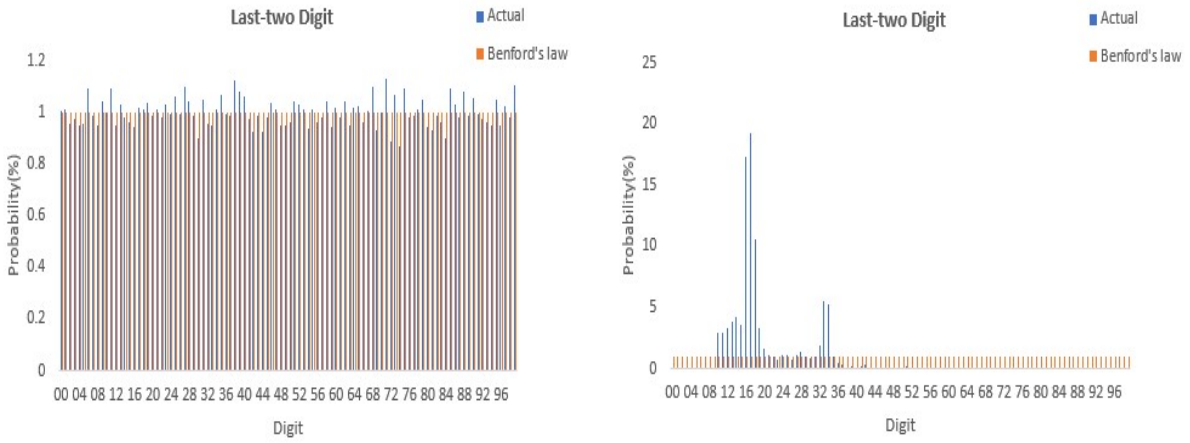
Figure D.2: Comparing TDT benign and zero-day network traffic on the UNSW-NB15 data set



(a) Benign network traffic.

(b) Malicious zero-day network traffic.

Figure D.3: Comparison of F2DT benign and zero-day network traffic on the UNSW-NB15 data set

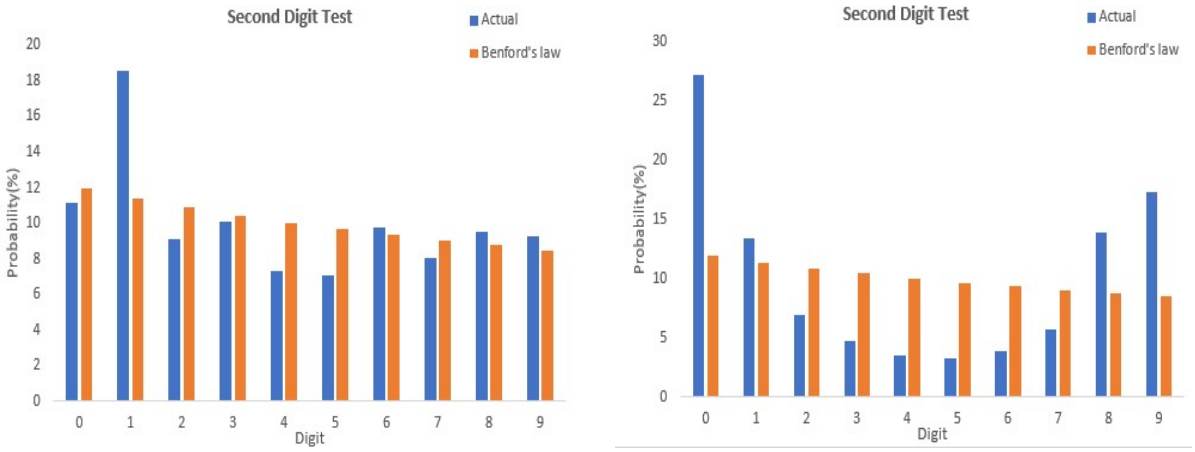


(a) Benign network traffic.

(b) Malicious zero-day network traffic.

Figure D.4: Comparing L2DT benign and zero-day network traffic on the UNSW-NB15 data set

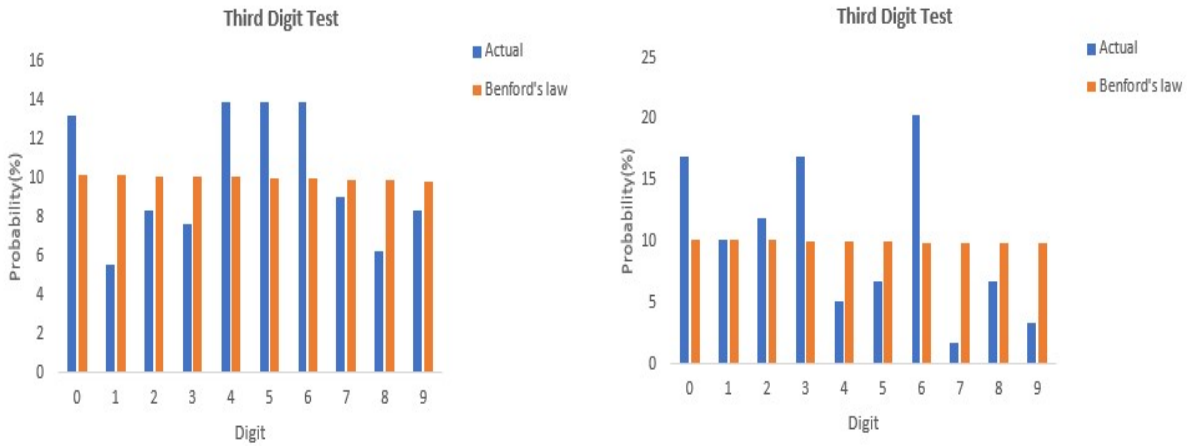
D.2 CICDDOS2019 Feature selection results



(a) Benign network traffic.

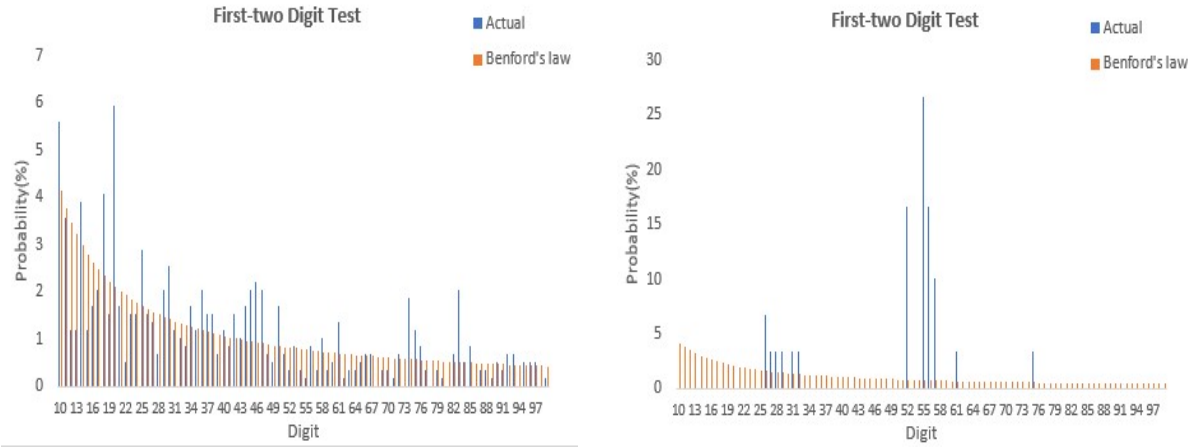
(b) Malicious zero-day network traffic.

Figure D.5: Comparing the SDT benign and zero-day network traffic on CICDDOS2019 data set



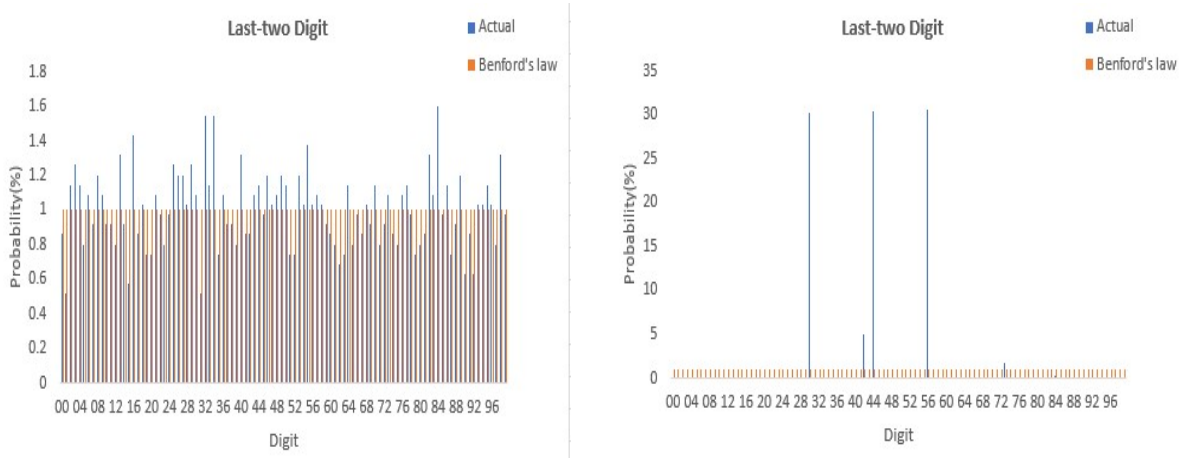
(a) Benign network traffic. (b) Malicious zero-day network traffic.

Figure D.6: Comparing TDT benign and zero-day network traffic on the CICDDOS2019 data set



(a) Benign network traffic. (b) Malicious zero-day network traffic.

Figure D.7: Comparing F2DT benign and zero-day network traffic on the CICDDOS2019 data set

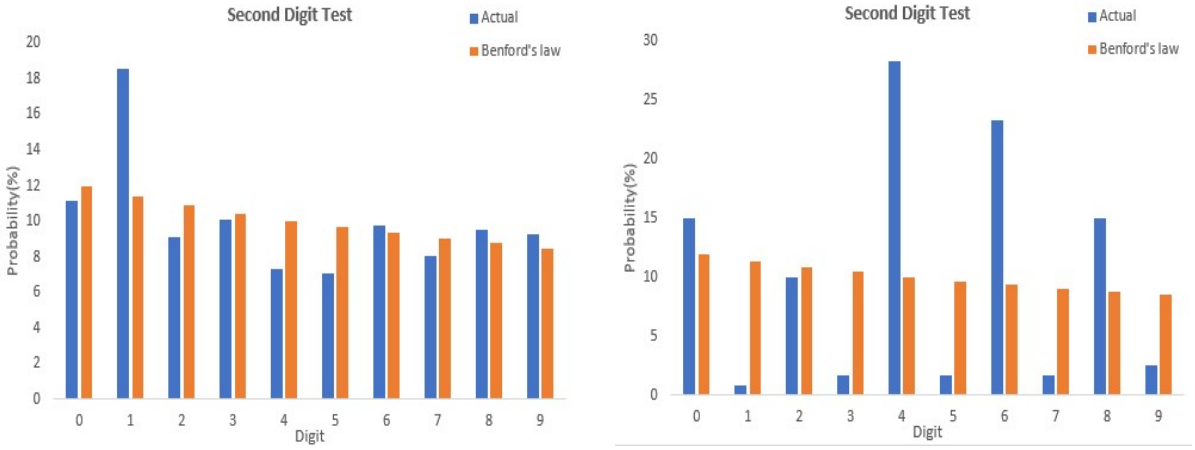


(a) Benign network traffic.

(b) Malicious zero-day network traffic.

Figure D.8: Comparing L2DT benign and zero-day network traffic on the CICDDOS2019 data set

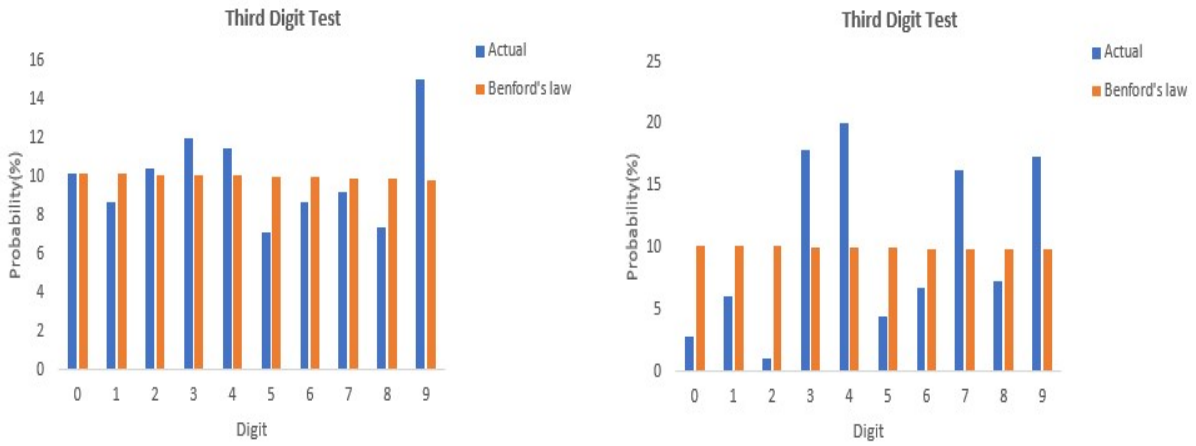
D.3 IoT Intrusion-2020 Feature selection results



(a) Benign network traffic.

(b) Malicious zero-day network traffic.

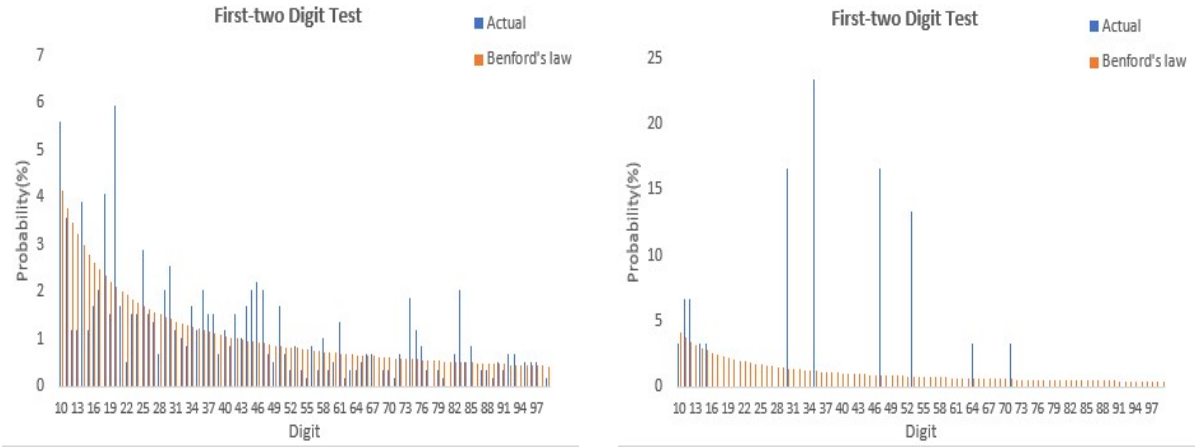
Figure D.9: Comparing SDT benign and zero-day network traffic on the IoT Intrusion-2020 data set



(a) Benign network traffic.

(b) Malicious zero-day network traffic.

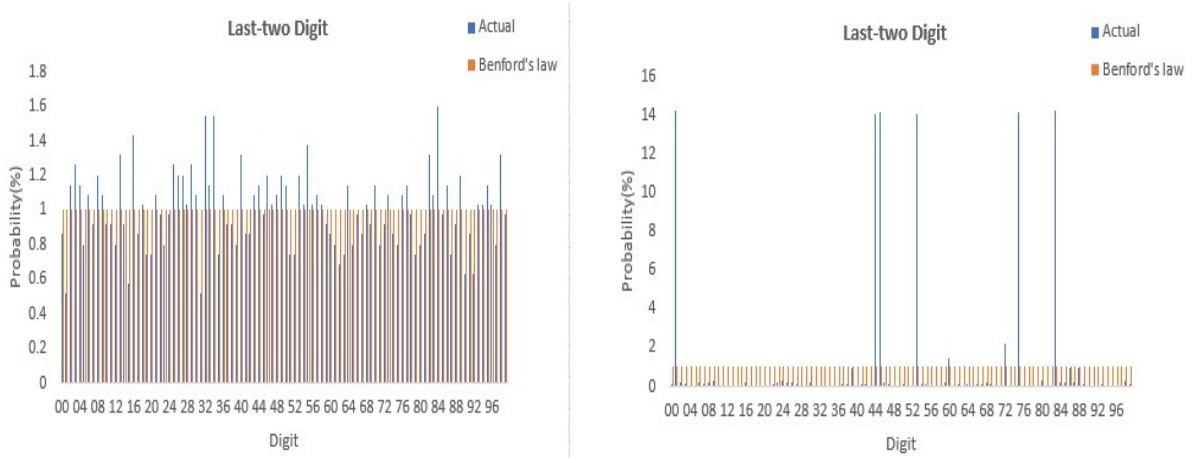
Figure D.10: Comparing TDT benign and zero-day network traffic on the IoT Intrusion-2020 data set



(a) Benign network traffic.

(b) Malicious zero-day network traffic.

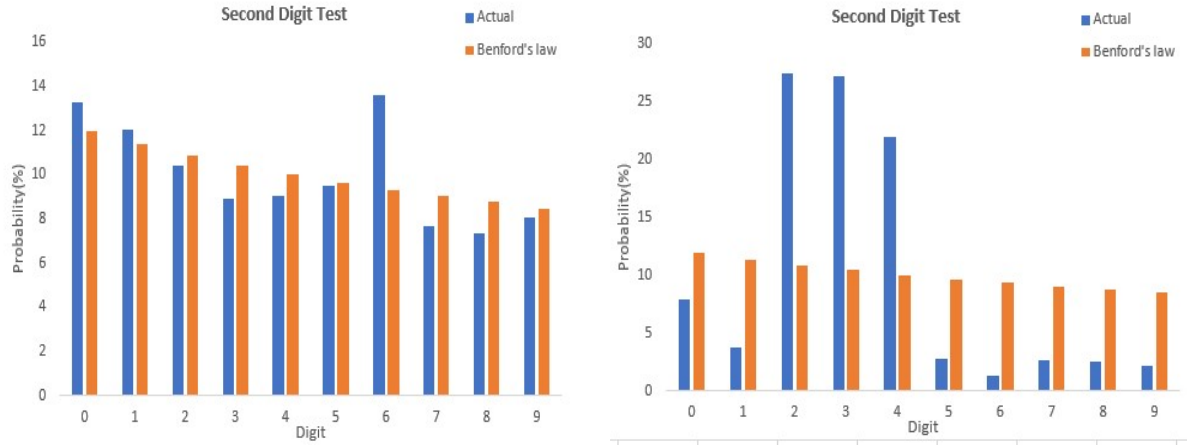
Figure D.11: Comparing F2DT benign and zero-day network traffic on the IoT Intrusion-2020 data set



(a) Benign network traffic. (b) Malicious zero-day network traffic.

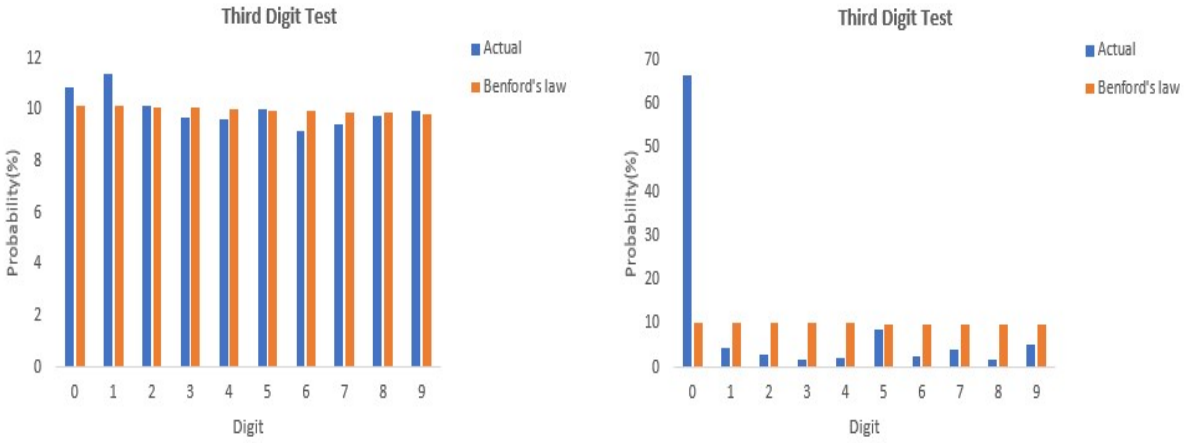
Figure D.12: Comparing L2DT benign and zero-day network traffic on the IoT Intrusion-2020 data set

D.4 CIRACICDOHBRW2020 Feature selection results



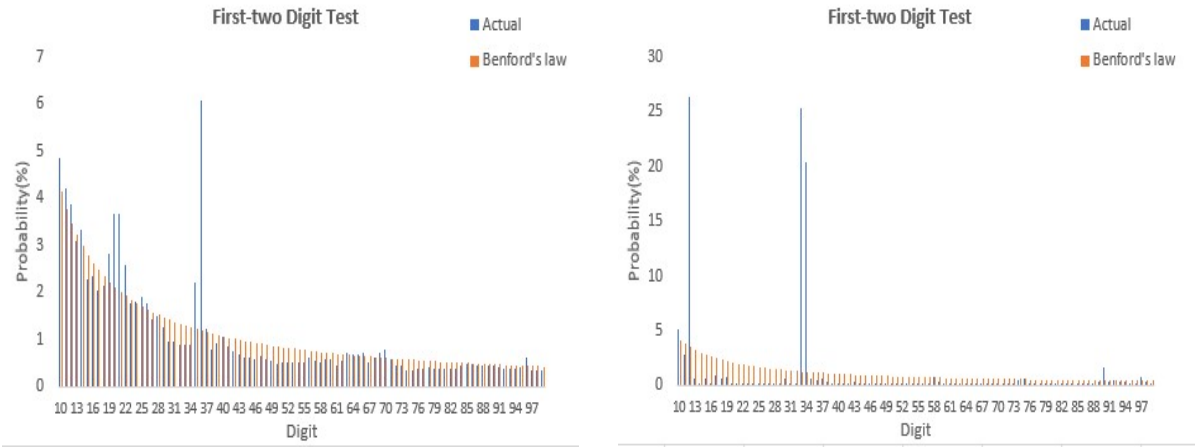
(a) Benign network traffic. (b) Malicious zero-day network traffic.

Figure D.13: Comparing SDT benign and zero-day network traffic on the IoT Intrusion-2020 data set



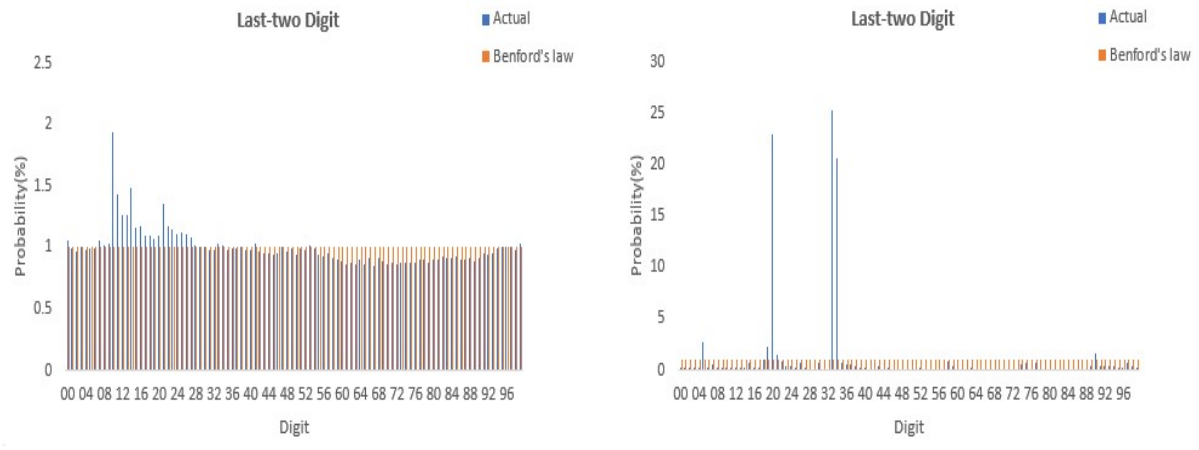
(a) Benign network traffic. (b) Malicious zero-day network traffic.

Figure D.14: Comparing TDT benign and zero-day network traffic on the IoT Intrusion-2020 data set



(a) Benign network traffic. (b) Malicious zero-day network traffic.

Figure D.15: Comparing F2DT benign and zero-day network traffic on the IoT Intrusion-2020 data set



(a) Benign network traffic.

(b) Malicious zero-day network traffic.

Figure D.16: Comparing L2DT benign and zero-day network traffic on the IoT Intrusion-2020 data set

Appendix E

CySecML methodology for discovering zero-day network intrusion and social engineering attacks

E.1 NIDS cybersecurity data sets for discovering zero-day network intrusion attacks

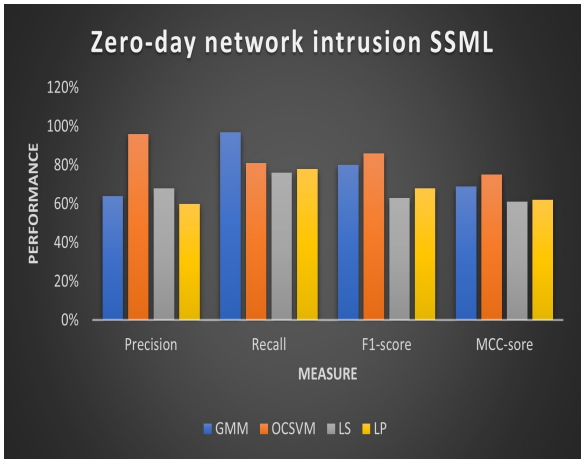
This section presents the InternetBotDetector model results for discovering zero-day network intrusion attacks using different data sets.

E.1.1 Discovering zero-day network intrusion attacks using the UNSW-NB15 data set

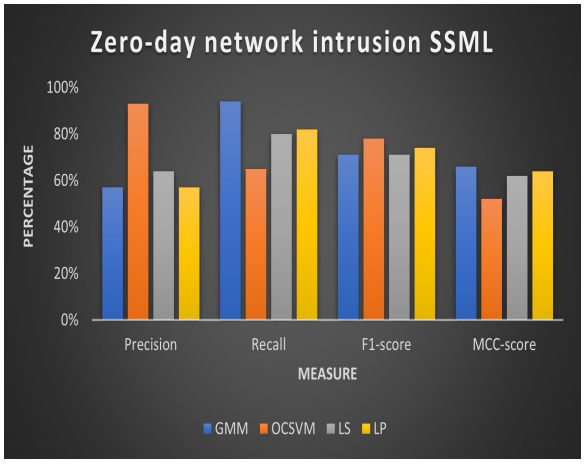
This section contains the results for discovering zero-day network intrusion attacks by using significant features identified by the BL method versus Sharafaldin et al. [9] as shown in Table 7.4.

SSML	Precision (%)		Recall (%)		F_1 (%)		MCC (%)	
	BL	[2]	BL	[2]	BL	[2]	BL	[2]
GMM	64	57	97	94	78	71	69	66
OCSVM	96	93	81	65	86	70	75	52
LS	68	64	76	80	63	71	61	62
LP	60	66	78	81	68	72	62	64

Table E.1: Discovering zero-day network intrusion attacks using the UNSW-NB15 data set for Experiments 1 and 2



(a) SSML zero-day discovery - Experiment 1



(b) SSML zero-day discovery - Experiment 2

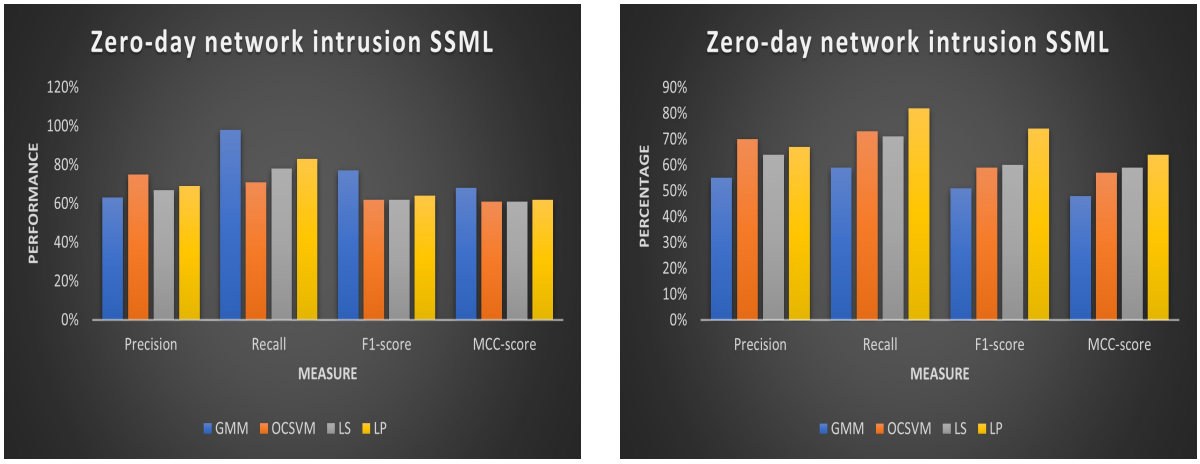
Figure E.1: SSML performance using the UNSW-NB15 data set for zero-day discovery

E.1.2 Discovering zero-day network intrusion attacks using the CICD-DOS2019 data set

This section contains the results for discovering zero-day network intrusion attacks by using significant features identified by the BL method versus Sharafaldin et al. [9] as shown in Table 7.3.

SSML	Precision (%)		Recall (%)		F_1 (%)		MCC (%)	
	BL	[9]	BL	[9]	BL	[9]	BL	[9]
GMM	63	55	98	59	77	51	68	48
OCSVM	75	70	71	73	62	59	61	57
LS	67	64	78	71	62	60	61	59
LP	69	67	83	82	64	74	62	64

Table E.2: Discovering zero-day network intrusion attacks using the CICDDOS2019 data set for Experiments 1 and 2



(a) SSML zero-day discovery - Experiment 1

(b) SSML zero-day discovery - Experiment 2

Figure E.2: SSML performance using the CICDDOS2019 data set for zero-day discovery

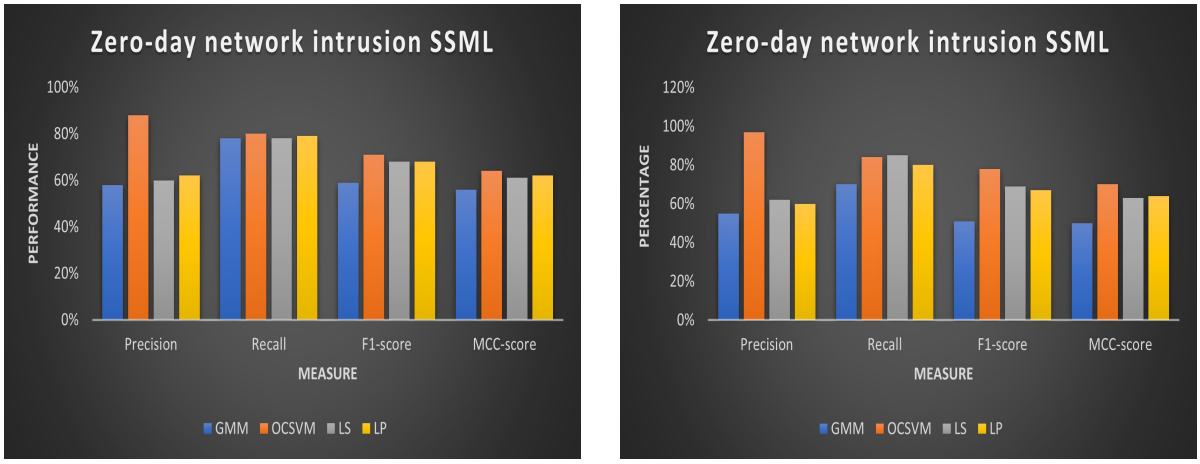
The results in Table E.2 and Figure E.2 demonstrate that the overall performance of the InternetBotDetector model is better when using significant features identified by the BL method than when using those identified by the authors [9]. The OCSVM SSML achieved the best overall results.

E.1.3 Discovering zero-day network intrusion attacks using the IoT Intrusion-2020 data set

This section presents results for discovering zero-day network intrusion attacks using significant features identified by the BL method versus Ullah .[8] as shown in Table 7.2.

SSML	Precision (%)		Recall (%)		F_1 (%)		MCC (%)	
	BL	[8]	BL	[8]	BL	[8]	BL	[8]
GMM	58	55	78	70	59	51	56	50
OCSVM	88	97	80	84	71	78	64	70
LS	60	62	78	85	68	69	61	63
LP	62	60	79	80	68	67	62	64

Table E.3: Discovering zero-day network intrusion attacks using the IoT Intrusion-2020 data set for Experiments 1 and 2



(a) SSML zero-day discovery - Experiment 1

(b) SSML zero-day discovery - Experiment 2

Figure E.3: SSML performance using the IoT Intrusion-2020 data set for zero-day discovery

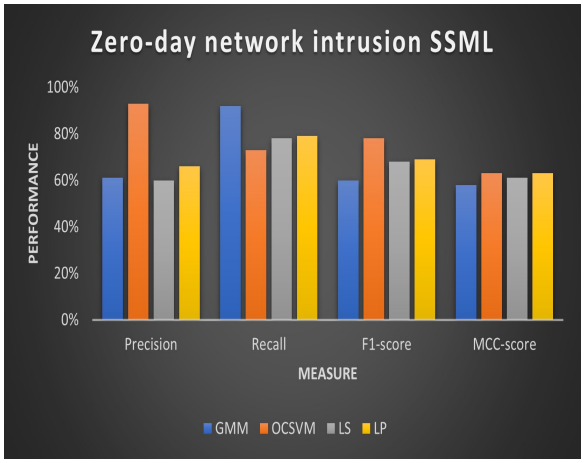
The results in Table E.3 and Figure E.3 demonstrate that the overall performance of the InternetBotDetector model is better when using significant features identified by the BL method than when using those identified by the authors [8]. The OCSVM SSML achieved the best overall results.

E.1.4 Discovering zero-day network intrusion attacks using the CIRACI-CDOHBRW2020 data set

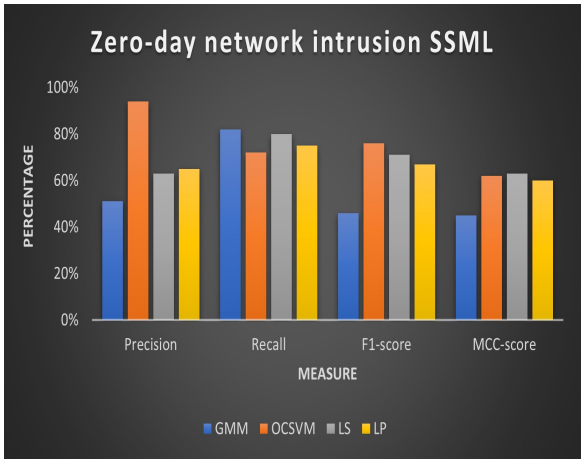
This section presents results for discovering zero-day network intrusion attacks by using significant features identified by the BL method versus Montazerishatoori .[10] as shown in Table 7.5.

SSML	Precision (%)		Recall (%)		F_1 (%)		MCC (%)	
	BL	[10]	BL	[10]	BL	[10]	BL	[10]
GMM	61	51	92	82	60	46	58	45
OCSVM	93	94	73	72	78	76	63	62
LS	60	63	78	80	68	71	61	63
LP	66	65	79	75	69	67	63	60

Table E.4: Discovering zero-day network intrusion attacks using the CIRACICDO-HBRW2020 data set for Experiments 1 and 2



(a) SSML zero-day discovery - Experiment 1



(b) SSML zero-day discovery - Experiment 2

Figure E.4: SSML performance using the CIRACICDOHBRW2020 data set for zero-day discovery

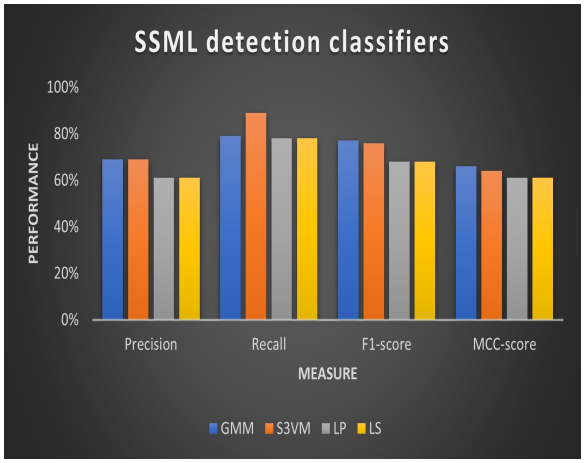
The results in Table E.4 and Figure E.4 demonstrate that the overall performance of the InternetBotDetector model is better when using significant features identified by the BL method than when using those identified by the authors [10]. The OCSVM SSML achieved the best overall results.

E.2 OSN cybersecurity data sets for discovering social engineering attacks

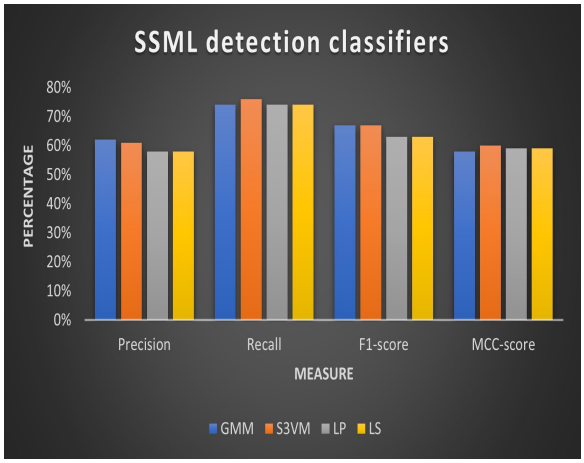
This section presents the InternetBotDetector model results for discovering social engineering attacks.

SSML	Precision (%)		Recall (%)		F_1 (%)		MCC (%)	
	BL	Table 5.6	BL	Table 5.6	BL	Table 5.6	BL	Table 5.6
GMM	69	62	79	73	61	67	60	58
S3VM	74	61	70	68	71	67	61	60
LP	61	58	78	74	68	63	61	59
LS	61	58	78	74	68	63	61	59

Table E.5: Discovering social engineering attacks for Experiments 3 and 4



(a) SSML discovery of social engineering attacks - Experiment 3



(b) SSML discovery of social engineering attacks - Experiment 4

Figure E.5: SSML performance for discovering social engineering attacks