UNIVERSITY OF SOUTHAMPTON
Faculty of Engineering and Physical Science
School of Electronics and Computer Science

A project report submitted for the award of
MEng Computer Science with Cyber Security

Project Supervisor: Dr. Enrico Gerding
Second Examiner: Dr. Huajie Yi

Exploring the Trade-off Between Model Complexity and Accuracy in MANET Security Using SVM

by Rishabh Arora

September 2025

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND PHYSICAL SCIENCE
SCHOOL OF ELECTRONICS AND COMPUTER SCIENCE

A project report submitted for the award of MEng Computer Science with Cyber Security

by Rishabh Arora

This research presents a comprehensive study on the application of machine learning techniques for network intrusion detection, focusing on the UNSW-NB15 dataset—a widely recognized benchmark in cybersecurity research. The project aimed to develop and validate predictive models capable of detecting and classifying network intrusions effectively. Using a combination of feature extraction, data preprocessing, and advanced machine learning algorithms, the study explored the impact of model complexity on detection accuracy. Various machine learning models, including Support Vector Machines (SVM) and decision trees, were evaluated to ascertain their effectiveness in distinguishing between benign and malicious network activities. The training and testing phases were carefully managed to ensure robust model evaluation, leveraging a balanced dataset approach to address class imbalance prevalent in network security datasets. The findings underscore the importance of feature selection and model tuning in enhancing detection accuracy, providing insights into the trade-offs between model complexity and performance in real-world intrusion detection scenarios. This study contributes to the field by offering a detailed methodological framework that can be employed to improve intrusion detection systems in both academic and practical cybersecurity applications.

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

You must change the statements in the boxes if you do not agree with them.

We expect you to acknowledge all sources of information (e.g. ideas, algorithms, data) using citations. You must also put quotation marks around any sections of text that you have copied without paraphrasing. If any figures or tables have been taken or modified from another source, you must explain this in the caption and cite the original source.

I have acknowledged all sources, and identified any content taken from elsewhere.

If you have used any code (e.g. open-source code), reference designs, or similar resources that have been produced by anyone else, you must list them in the box below. In the report, you must explain what was used and how it relates to the work you have done.

I have not used any resources produced by anyone else.

You can consult with module teaching staff/demonstrators, but you should not show anyone else your work (this includes uploading your work to publicly-accessible repositories e.g. Github, unless expressly permitted by the module leader), or help them to do theirs. For individual assignments, we expect you to work on your own. For group assignments, we expect that you work only with your allocated group. You must get permission in writing from the module teaching staff before you seek outside assistance, e.g. a proofreading service, and declare it here.

I did all the work myself, or with my allocated group, and have not helped anyone else.

We expect that you have not fabricated, modified or distorted any data, evidence, references, experimental results, or other material used or presented in the report. You must clearly describe your experiments and how the results were obtained, and include all data, source code and/or designs (either in the report, or submitted as a separate file) so that your results could be reproduced.

The material in the report is genuine, and I have included all my data/code/designs.

We expect that you have not previously submitted any part of this work for another assessment. You must get permission in writing from the module teaching staff before re-using any of your previously submitted work for this assessment.

I have not submitted any part of this work for another assessment.

If your work involved research/studies (including surveys) on human participants, their cells or data, or on animals, you must have been granted ethical approval before the work was carried out, and any experiments must have followed these requirements. You must give details of this in the report, and list the ethical approval reference number(s) in the box below.

My work did not involve human participants, their cells or data, or animals.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to first thank Dr. Enrico Gerding for his continuous support through this project. His advice has made a big difference in my work.

I also want to give heartfelt thanks to PhD researcher, Charles Hutchins for providing invaluable guidance on topics in this project. Without his help, this project would not have come to fruition. Last but not the least, I want to thank my coursemates Shantam Sridev, Sunnie Jehan-Morrison and Josh Pattman for supporting in this journey.

Chapter 1

Introduction

This Introduction outlines the critical aspects of security in Mobile Ad-Hoc Networks (MANETs). Divided into three main parts— **Motivation**, **Challenge**, and **Goals**, the following chapter provides an overview of the initial objectives of this project.

1.1 Motivation

Mobile Ad-Hoc Networks (MANETs) are vital in situations where conventional network infrastructures are impractical, such as military operations and emergency rescues. Decentralised nature and dynamic topology, securing MANETs against threats such as grayhole attacks. A grayhole attack is a type of network threat where a malicious node selectively drops packets it agrees to forward, making it particularly challenging to detect because it behaves normally at other times. The study focuses on the resource limitations that affect the efficiency of Flying Ad-Hoc Networks (FANETs), such as short battery life and low processing power [Wang and Jiang, 2022]. It looks at ways to maximise performance within these constraints, by using better power management and routing protocols. Its goal is to make security systems in MANETs work better and more efficiently by using intelligent feature selection and support vector machines (SVMs).

1.2 Challenge

A large number of features were involved in these ML models Sivanesan et al. [2023], Chourasia and Tokekar [2024], Dadi and Abid [2022], Sivanesan and Archana [2022], Alheeti et al. [2016b], and we can hypothesise that we can have a very similar accuracy with fewer number of features. Because there are so many features within these models,

it will be good to investigate the contribution of each feature to accuracy.

1.3 Goals

The primary objectives are mentioned below. The completion of these goals will mark the success of this project.

1. **Simulation Development:**
   The initial objective of this study is to design and implement NS3 environment that can generate virtual networks in order to simulate scenarios that are unique to the dynamics of MANETs. This will enable the generation of network traffic data that can be exported as packet capture files or transformed into datasets suitable for analysis.

2. **Feature Extraction:**
   Following the simulation, a key goal is to extract meaningful features and convert them into datasets. A thorough review of existing research will serve as the basis for choosing these features.

3. **Training SVM Model:**
   Taking the dataset into consideration, another goal is to train a robust SVM model. The purpose of this model is to analyse and assess the influence of different network characteristics on the accuracy of grayhole attack detection. The aim is to identify critical characteristics that optimise resource utilisation while maintaining a high level of accuracy.

4. **Feature Analysis:**
   In conclusion, this study's ultimate objective is to provide a comprehensive analysis of how particular features influence the SVM model's precision. This will include an exploration of the contribution of each feature to the overall effectiveness of the detection systems.

These efforts are directed towards addressing specific technical challenges in MANETs and enumerating the application of ML in network security. The anticipated conclusions are expected to provide actionable insights that could significantly enhance the robustness and efficiency of security mechanisms in MANETs and potentially other types of Ad-hoc networks.

Chapter 2

Background and Related Work

This chapter reviews key literature and foundational concepts pertinent to MANETs, focusing on influential research and prevailing challenges.

2.1 Background Literature

This section entails the background research that was conducted to understand the scope of the project. It covers MANETs Networks, Greyhole Attacks and Support Vector Machines (SVMs). Each of these topics set the fundamental base of the study undertaken.

2.1.1 MANET Network

Mobile Ad-Hoc Networks (MANETs) are dynamic, decentralised networks. In contrast to traditional networks, where nodes retrieve data from a central server. MANETs use peer-to-peer (P2P) networking. MANETs use wireless connectivity to connect mobile nodes. MANETs build networks on the fly without central control, making them unique.

Every MANET node is a host and a router. Thus, it may transmit, receive, and forward packets to surrounding nodes. Nodes freely enter and leave the network, changing the topology. This dynamic nature alters packet delivery. Because of this, every node must constantly update its routing data to find network paths [Gupta et al., 2018].

2.1.2 Greyhole Attack

MANETs work on routing protocols like AODV (Ad-hoc On-Demand Distance Vector) and DSR (Dynamic Source Routing), which are responsible for the exploration and

maintenance of routes between nodes. These protocols are based on the idea that nodes should trust each other to forward packets [Perkins and Royer, 1999].

A Greyhole Attack exploits these vulnerabilities by breaking the trust that nodes have in each other within the MANET. In this attack, a malicious node behaves like a benign node in the network, participating in the packet forwarding process and following protocol rules to earn trust with other nodes. However, unlike a straightforward blackhole attack where the malicious node drops all the packets passing through, it drops at a specific time or can be probabilistic, so you can drop 50% or 60% of the time, making it much more challenging to detect [Singh et al., 2021].

Understanding and mitigating such attacks is important for maintaining the reliability and security of MANET networks. There have been attempts where various methods are used to defend against these attacks, like reinforcement learning as mentioned in [Chourasia and Tokekar, 2024] and game theory as mentioned in [Hutchins et al., 2024], but one of them is support vector machines, which we are going to talk about in this paper.

2.1.3 Support Vector Machine

In the context of MANETs, SVMs can be extremely useful for addressing security issues, such as detecting and mitigating grayhole attacks. Given the dynamic and decentralised MANETs, the network constantly requires robust mechanisms to ensure security and reliability while transferring packets among nodes. SVMs can be trained on features of network traffic to identify patterns that violate the standards and are suspected of performing malicious activity.

2.2 Existing Research

The specialized nature of MANET networks require a thorough scout of previous research conducted by researchers. The following subsections explore pivotal studies that inform the approach relating to dataset and features.

2.2.1 Feature Contribution

Numerous studies have been conducted, each of which has developed an Intrusion Detection System (IDS) based on SVMs for the purpose of identifying malicious nodes within MANET networks. These studies also listed the features that were used, but they did not talk about how each important feature contributed [Alheeti et al., 2016a, Dadi and Abid, 2022, Poongothai and Duraiswamy, 2015].

Dadi and Abid [2022] proposes a novel approach to secure the Internet of Vehicles (IoV) against a range of attacks through AutoEncoder (AE) and SVMs. Using the UNSW-NB15 dataset, the authors proved that their system could find attacks such as Denial of Service (DoS), Distributed Denial of Service Attack (DDoS), Wormhole, Blackhole, and Greyhole attacks. It did this with 49 features and a reported accuracy of 96.7%.

While the paper effectively Portrays the utility of merging AE and SVM for intrusion detection, it notably lacks a detailed discovery of the individual contributions of major features used towards the model's overall accuracy. The authors claim a robust detection capability, yet there is no discussion about the percentage contribution or major features, which leaves us questioning the relevance and weight of specific features in the system's performance. This omission is significant as it masks which features might be redundant or critical.

Alheeti et al. [2016a] provides a comprehensive methodology to mitigate greyhole attacks and rushing attacks in self-driving vehicular networks. The study uses features taken from the Network Simulator 2 (NS2) trace file to support a combination of feed-forward neural networks and SVMs.

The research demonstrates a high accuracy of 99.71% with the help of 15 chosen features, which is better than Dadi and Abid [2022] with a reduced number of features and with reduced false alarms. However, this paper also doesn't discuss the individual contribution of each of the employed features towards the model's accuracy. This hides the significance as it prevents understanding which features are most influential in detecting attacks and which may be redundant. Such information is important for refining the model to enhance its efficiency and adaptability to different and evolved attack vectors.

Another study by [Poongothai and Duraiswamy, 2015] presents a novel method that combines Rough Set Theory (RST) and Support Vector Machines (SVM) to enhance the precision of detection in MANETs. There is not enough information about how many features were kept post-RST. This makes it harder to reproduce the model and fully understand how well it works. To thoroughly evaluate and apply this, future research should focus on filling this gap by offering precise metrics on the processes of feature selection and reduction.

2.2.2 Feature Extraction Methods

The research by Alheeti et al. [2016b] commences with the NS2 simulator executing scenarios that replicate diverse traffic and mobility conditions characteristic of MANET environments. Trace files are generated during these simulations to record comprehensive information about network activities, including packet transmissions and node movements.

Specific features are meticulously chosen for extraction from these trace files. The significance of the data in relation to the IDS's requirements—which may include specifics like packet sizes, transmission times, and source and destination addresses—determines the selection process. The chosen characteristics are critical for the system to accurately identify and examine network behavior.

After identifying the relevant features, they are made ready for integration into the IDS. This process entails organising the data into a format that is compatible with the detection algorithms and potentially includes preprocessing steps like data normalisation. Normalisation is a process that makes measurement scales consistent across different features, improving anomaly detection accuracy by standardizing them.

This methodical approach to extracting features from NS2 trace files ensures that the IDS is provided with data of exceptional quality and relevance. This is crucial for the system's capacity to accurately detect and react to potential security risks within MANETs. The careful extraction and organization of data serve as the foundation for the system's ability, making it an essential element in upholding network security.

Another approach was conducive by Dadi and Abid [2022] in which they used feature extraction to improve the data for a more advanced IDS. This IDS combines the abilities of AE and SVMs. The study's methodology starts by using the UNSW-NB15 dataset, which includes a diverse set of network traffic characteristics. This dataset forms the basis for identifying and preparing the necessary data features.

Autoencoders are primarily used in the feature extraction phase to compress and encode raw data into a more manageable and indicative set of features. This step is essential because it decreases the number of dimensions in the dataset while preserving the important characteristics needed to identify potential network threats. The compressed features, which have been refined and intensified, are then analysed using Support Vector Machines (SVM). This phase utilises the robust classification capability of SVM to accurately differentiate between benign and malicious network behaviours.

Using AE and SVMs in the feature extraction process is important because it lets complicated data structures be processed and helps find small patterns that point to problems in the network. This methodology not only simplifies the process of analysing data but also improves the ability of the IDS to accurately predict and address potential security risks in network environments.

The careful method of extracting features, which focuses on how deep learning techniques and statistical models can work together, provides a complete framework for building strong intrusion detection systems. The text highlights the significance of employing advanced data processing techniques to improve the effectiveness of network security systems.

Different study was conducted by Zardari et al. [2019] on feature extraction in MANETs using IDS to specifically detect black and grey hole attacks. The process commences by deploying IDS nodes that are chosen using the Connected Dominating Set (CDS) technique. These nodes are strategically chosen to maximize surveillance coverage throughout the network.

After being deployed, these Intrusion Detection System (IDS) nodes play a vital role in collecting data by transmitting status packets. This activity allows them to collect up-to-date information about the network's behaviour, which is crucial for extracting important characteristics. The collected data encompasses diverse metrics, including packet counts, sequence numbers, and transmission times, which are relevant to the operational status of the network.

By analysing this gathered data, distinct characteristics are extracted that play a crucial role in identifying abnormal behaviour that suggests possible security risks. The extraction process aims to detect anomalies and patterns in the data that deviate from normal behavior, indicating the possibility of black or grey hole attacks. This entails examining packet transmission patterns and identifying sequence number inconsistencies, which are clear indications of such attacks. The efficacy of this process of extracting features is vital, as it directly impacts the precision and effectiveness of the IDS in identifying and reacting to potential threats. In MANET, the IDS can effectively protect against advanced network attacks by selectively monitoring and analyzing specific data points. This focused method of extracting features guarantees that the security measures are responsive and anticipatory, thereby strengthening the overall robustness of the network.

2.2.3 UNSW-NB15 Dataset

The UNSW-NB15 dataset is designed to accurately represent current network behaviours and attack scenarios, as well as new attack methods that are not included in older datasets like KDD98 or KDDCUP99. The research's contemporary relevance ensures that it is firmly based on current security challenges, thereby increasing its applicability and effectiveness in modern systems. Extensive Range of Features: This dataset contains a diverse set of descriptive features obtained from network traffic using sophisticated tools such as Argus and Bro-IDS. There are many of these traits, such as detailed flow statistics and accurate attack signatures, that make it possible to build advanced intrusion detection models that can find both simple and complex threats. In comparison to its previous versions, UNS-NB15 exhibits a wider range of attack vectors, showcasing a greater diversity of attack types. The software incorporates simulations of contemporary forms of attacks, such as shellcode, backdoors, and worms. These simulations are essential for evaluating the effectiveness of intrusion detection systems in detecting and defending against the most recent threats. In order to verify the efficacy of IDS models in different and difficult circumstances, the presence of diversity is critical.

Chapter 3

Methodology

The study's fundamental methodology commenced with an initial simulation of a MANET network using Network Simulator 3(NS3) [The ns-3 Network Simulator Project]. While observing NS3 simulations, it was highlighted that the specific existing datasets like KDD Cup 99 and NSL-KDD were limited, particularly their inability to represent the dynamic and mobile nature of MANETs adequately. It can also be argued that these datasets don't contain modern-day attack strategies. Progress was made towards dataset creation where a packet network was created and 2 tracing methods: using Packet Capture (pcap) files and Trace Files were attempted. The UNSW-NB15 dataset was then selected to continue with the project. Next, SVM models were trained based on this dataset to facilitate predictions and subsequently cleanse the dataset. In the following section, we will determine the importance of major features in improving the model's precision and examine the most impactful features.

3.1 Network Simulation

In this project, the first challenge involves simulating a network using NS3. NS3, a modular framework in C++, enables detailed simulations of network scenarios, ensuring realism and scalability. NS3 is capable of extending far beyond mere simulations. It serves as a comprehensive toolkit for modelling, analysing, and optimising diverse network architectures with precision and flexibility.

3.1.1 Configuration

The computational experiments were conducted on a computing system that was outfitted with Radeon Vega Mobile Gfx-enabled AMD Ryzen 5 3500U processor clocked at 2.1 GHz (base) to 3.7 GHz (boost), in addition to 16 GB of DDR4 RAM. The AMD

Radeon Vega 8 Graphics was the integrated graphics processing unit. The system's storage consisted of a 736.2 GB HDD. The hardware ran on a software platform that included GNOME 42—9 for the desktop environment and Ubuntu 22.04.3 LTS with a 64-bit architecture. The NS3.40 software framework is taken into consideration for NS3 simulations.

3.1.2 Point to Point Network

The research began by setting up a basic point-to-point network to explore the NS3 simulator's functionality and reliability. This initial stage was crucial for confirming the correct operation of network communications and the accuracy of the NS3 models. The study ensured the correct implementation of simulation scripts and network protocols by starting with a simpler network. This foundational work also offered a valuable understanding of the NS3 simulator's capabilities and limitations, vital for creating more complex network simulations. Figure 3.1 is a screenshot from NetAim, which is a software built into NS3 that is capable of showcasing the movement of nodes and packet transmission within the simulated network.

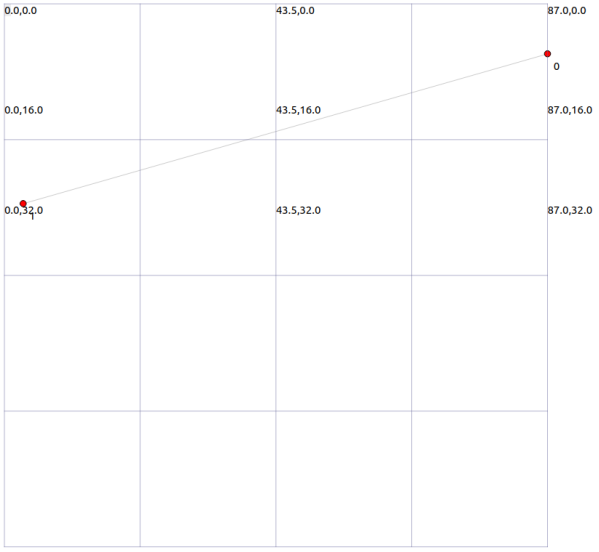

FIGURE 3.1: Screenshot captured from NetAnim illustrating a point-to-point network configuration

3.1.3 MANET Network

The next step in the research was to design a replica of the MANET scenario. The MANET network consisted of **21 benign nodes** and **3 malicious nodes** that were dispersed across a designated region. Modelling the mobility of nodes at a rate of **10m/s**, the Random Waypoint Mobility Model was employed. The simulation was defined as **100m x 100m**, allowing nodes to move and interact as dynamic nodes.

To simulate wireless communication, the standard Wi-Fi module from NS3 was used. To simulate real-world conditions, the physical layer parameters, such as transmission power levels, were set to **7dBm** and data rates were configured to **2048bps**. The use of a constant-rate Wi-Fi manager ensured that all nodes transmitted data at the same rate.

The **Ad hoc On-Demand Distance Vector (AODV)** routing protocol was utilised to manage routing in the MANET. Custom modifications were implemented to simulate attack behaviour for evaluation. To simulate malicious behaviour, the researchers implemented a custom AODV variant that incorporated a grey attack strategy. This was achieved by extending the AODV protocol and introducing additional parameters to regulate packet-dropping behaviour.

Receiving sockets were established on the nodes by the simulation script in order to capture and analyze packet transmissions. Every individual node was furnished with a UDP socket designed to receive packets. This configuration allowed packet delivery and network performance metrics, such as packet loss and throughput, to be evaluated.

The OnOff application module was utilised to simulate network traffic by generating UDP packets at a predetermined data rate. The benign nodes were utilised to distribute packet generation sources, thereby simulating communication with external entities.

Enabling the tracing and animation features in NS3 allowed for the simulation to be visualized. We employed NetAnim to visually represent the real-time movement of nodes and packet transmissions. The FlowMonitor module from NS3 was also used to collect performance data. This lets the dependability and effectiveness of the AODV routing protocol be tested in a variety of network situations.

Table 3.1 presents a more detailed configuration of the simulation environment used for the experiments.

Parameter	**Value**
Simulator	ns-3.40
Simulation time	200 s
Number of nodes	24 (21 benign, 3 malicious)
Topology	100 x 100 (m)
Speed	10 m/s
Mobility Models	RandomWaypointMobilityModel
Type of Traffic	UDP (OnOffApplication)
Packet Size	800 bytes
Transport Protocol	UDP
Routing Protocol	AODV with Greyhole Attack
Channel type	YansWifiChannel
MAC protocol	IEEE 802.11b
Network Interface type	Physical Wireless

TABLE 3.1: Simulator Environmental and Parameters

3.1.4 Network Visualization

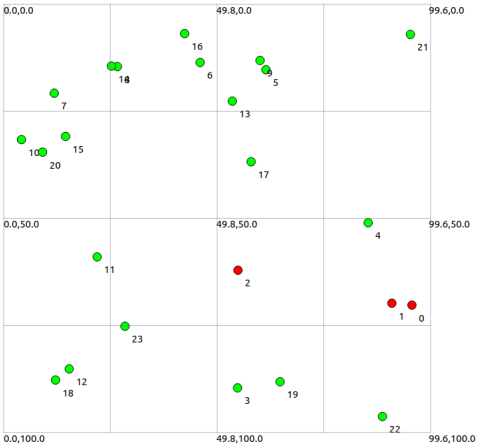

FIGURE 3.2: Screenshot captured from NetAnim illustrating MANET network at 0 sec

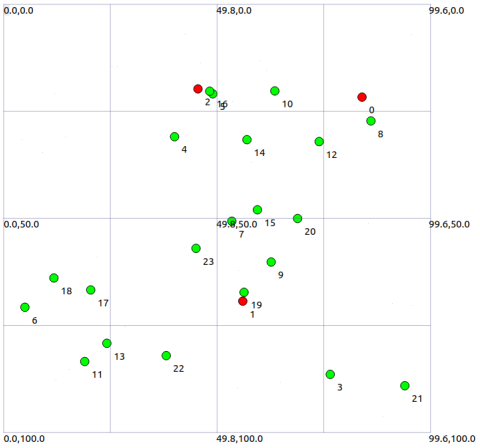

FIGURE 3.3: Screenshot captured from NetAnim illustrating MANET network at 50 sec

Moving forward, Figures 3.2 and 3.3, generated using NetAnim, depict the configured network settings. Figure 3.2 displays a snapshot of the NetAnim file at time t = 0 seconds, while Figure 3.3 demonstrates the state of the network at time t = 50 seconds. The visuals depict red nodes as malicious nodes, whereas green nodes represent benign nodes within the network. Notably, the screenshots document the ever-changing motion of nodes throughout the simulation.

3.2 Features

This section outlines the procedure for extracting features and converting them into a dataset that is appropriate for training the model. The methodology is structured into three subsections, each dedicated to a pivotal stage in the dataset creation process.

Section 3.2.1 examines prior studies to determine the crucial characteristics that are significant for the investigation. This step lays the foundation for the subsequent processes. A thorough examination of each feature is conducted to guarantee its accuracy and relevance, which is crucial for successful model training.

Section 3.2.2 details the process of extracting the identified features from the NS3 simulator albeit this direction was not developed further.

Section 3.2.3 talks about the reasoning behind the selection of the dataset to progress with the rest of the project.

3.2.1 Feature Identification

Following the implementation and validation of the MANET network within the NS3 environment, the primary focus of the next phase of the research was to identify critical features that were to be extracted. For this endeavour, a thorough review of the existing literature was required in order to identify key features that have been deemed significant in studies that are comparable to this one. The study performed a thorough identification of key features that are necessary for comprehensive network analysis.

In this research done by Alheeti et al. [2016b], the authors extracted a number of features from network simulation trace files in order to identify abnormal behaviours in vehicular networks. In spite of the fact that they narrowed their feature set down to 15 key features, the following specific details were provided:

Feature	**Description**
Packet ID	Unique identifier for each packet in the network
Payload Size	The size of the data within the packet
Payload Type	The type or category of the data payload
Source MAC Address	The MAC address of the packet's source node
Destination MAC Address	The MAC address of the packet's destination node
Ethernet Type	Protocol number to indicate the type of payload in the Ethernet frame
IP Source Address	The IP address of the packet's source
IP Destination Address	The IP address of the packet's destination
Packet Tagged Status	Indicator if the packet is tagged with VLAN or priority information
Hop Counts	The number of hops a packet has traversed
Broadcast ID	Identifier for broadcast messages to distinguish different sessions
Destination IP with Seq. Number	Combined information of the destination IP and the sequence number
Source IP with Seq. Number	Combined information of the source IP and the sequence number

TABLE 3.2: Description of Network Features

A larger dataset is used to train the IDS. The excerpt that has been provided does not, however, provide any specific information regarding the nature of the final 15 features that were chosen using the proportional overlapping scores (POS) method. The Proportional Overlapping Scores (POS) method is a statistical approach used to select features by measuring the degree of overlap between the distributions of each feature across different classes.

3.2.2 Feature Extraction

After identifying the key features, the next step was to extract them from the NS3 simulator. At this juncture in the project, it was researched and found that progressing with full feature extraction methods.

There are two commonly used methods for feature extraction; Used of Auto Encoders and Proportional Overlapping Scores (POS). The following explains why both these techniques were not viable to implement.

Auto encoders is an artificial neural network used primarily for unsupervised learning tasks, particularly for the purpose of data encoding and compression. It can also be used for feature extraction but they have various requirements and depth as explained in Dadi and Abid [2022] for usage in projects relating to MANETs. As explained in the paper the following reasons explain why AutoEncoder are not feasible:

- **Power Intensive**: Autoencoders require significant computational resources due to their complex structures and the large data volumes they process.
- **More Knowledge Required**: Effective use of autoencoders demands a deep understanding of neural networks, unsupervised learning, and domain-specific knowledge, such as intrusion detection techniques.
- **Complexity Increased**: The dual nature of autoencoders, involving both encoding and decoding processes, adds complexity to the modeling and computational process.
- **Overfitting Risk**: There's a risk of overfitting with autoencoders, especially if they're too complex for the amount of available data, which can impair their ability to generalize to new scenarios.
- **Expertise Required**: Deploying autoencoders effectively in applications like intrusion detection systems requires expertise in neural network architectures, data preprocessing, and integration with other machine learning models like SVMs.

POS was also a possible method but it again could not be implemented due to various technical reasons and depth of understanding required to use. As explained in paper Alheeti et al. [2016b], the following reasons are presented:

- **High Complexity and Resource Consumption**: The Proportional Overlapping Scores (POS) method, employed in the intrusion detection system (IDS), is complex and consumes significant power and memory due to intensive computation and data handling requirements.
- **Extraction of Unnecessary Information**: The methods utilized, like POS, might extract extraneous information, complicating the analysis and potentially cluttering the dataset with irrelevant data.
- **Risk of Overfitting**: Complex features derived from sophisticated methods such as POS can lead to overfitting, where the model learns the noise in the data rather than the actual signal, impairing its generalization to new data.
- **Scalability Issues**: The IDS described faces scalability challenges due to its reliance on resource-intensive processes and sophisticated algorithms, which may not perform efficiently as network size and data volume grow.

Another technique attempted was with the use of PCAP files to analyse the traffic packets in a network. In a typical home network, packets are transmitted directly from one node to another. A extensive resource was found (Luca Divit) for home network implementation of feature extraction however, in a MANET, the packet transmission involves multiple hops from one node to another. Consequently, the features extracted from resources designed for traditional network datasets were not effective for this MANET dataset and design of a completely new pcap file technique for MANETs from first principles was considered not to be a possibility for the itme frame of the project.

Additionally, the lack of centralized control in MANETs means that capturing a comprehensive view of the entire network through a single pcap file is impractical, if not impossible. The scarcity of online resources offering pcap analysis specific to MANETs further underscores the challenges and limitations of applying traditional home network analysis tools to ad-hoc networks, emphasizing the need for specialized tools designed to handle the dynamic nature of MANETs.

The lack of resources for feature extraction methods and simulation documentation for MANET networks necessitated the use of a predefined dataset to continue.

3.2.3 Dataset Selection

These papers were used as background research to fabricate the commercially available UNSW-NB15 dataset [Moustafa and Slay, 2015, 2016, Moustafa et al., 2017, 2019, Sarhan et al., 2021]. This section is based on the work of Moustafa and Slay [2015].

The UNSW-NB15 dataset was chosen for this research based on several compelling reasons that align with the goal of improving intrusion detection capabilities in modern network environments:

The dataset's high accessibility facilitates thorough testing and benchmarking in various research projects and methodologies, promoting a collaborative and iterative approach to enhancing IDS technologies. The dataset is easily accessible to the research community, allowing for easy reproduction and comparison in intrusion detection research. The dataset's high accessibility facilitates thorough testing and benchmarking in various research projects and methodologies, promoting a collaborative and iterative approach to enhancing IDS technologies. The dataset serves as both a means for creating novel detection techniques and a standard for assessing and contrasting the effectiveness of current intrusion detection models. The UNSW-NB15's dual functionality makes it an invaluable asset for both developmental and evaluative research in network security.

Because of these factors, the UNSW-NB15 dataset was chosen as the best and most appropriate one for research that aimed to improve the ability of intrusion detection systems to find and stop attacks in current and future network structures.

The dataset encompasses a range of attacks such as DoS, Exploits, and Worms, offering a comprehensive view of network vulnerabilities that grey hole attacks exploit. Studying these attacks helps develop targeted detection and mitigation strategies, enhancing the ability to identify and counteract grey hole tactics in diverse network environments.

3.3 Dataset Training

The UNSW-NB5 dataset consists of two million and 540,044 records, which are divided into four smaller datasets. A portion of this dataset has been made to Separated into four distinct subsets, the UNSW-NB15 dataset consists of 2,540,044 records overall. 175,341 records make up the part of these that has been designated as the training dataset. This department offers an organised method for training and assessing models, guaranteeing a thorough examination in a variety of network situations [Moustafa and Slay, 2015].

Table 3.3 provides a detailed summary of the UNSW-NB15 dataset's features, as well as an understandable explanation of each feature. In this comprehensive table, the 49 unique variables that make up the dataset are described in great detail. These characteristics include more complex metrics like packet sizes and inter-arrival times, in addition to basic network identifiers like IP addresses and port numbers. The dataset's structure makes it feasible to perform a detailed analysis of network traffic, which leads to a thorough understanding of patterns that are both potentially harmful and benign. The aforementioned attributes serve as the essential analytical basis for the instruction,

examination, and validation of sophisticated intrusion detection algorithms during the training phase.

Feature	**Description**
srcip	Source device's IP address.
sport	Port number used by the source device.
dstip	Destination device's IP address.
dsport	Port number used by the destination device.
proto	Type of protocol used for communication (like TCP or UDP).
state	Current state of the network connection (e.g., connected, closed).
dur	How long the network connection lasted.
sbytes	Total data in bytes sent from source to destination.
dbytes	Total data in bytes received from destination to source.
sttl	Remaining life of the packet from source to destination.
dttl	Remaining life of the packet from destination to source.
sloss	Packets lost from source.
dloss	Packets lost to destination.
Sload	Speed of data sent from the source (bits per second).
Dload	Speed of data received at the destination (bits per second).
Spkts	Number of packets sent from the source.
Dpkts	Number of packets received at the destination.
swin	Size of the last window advertised by the source.
dwin	Size of the last window advertised by the destination.
stcpb	TCP sequence number of the first packet from source.
dtcpb	TCP sequence number of the first packet to destination.
smeansz	Average size of packets sent by the source.
dmeansz	Average size of packets received by the destination.
$trans_depth$	The depth of the packet within the transaction process.
res_bdy_len	Length of the data payload in the packet.
Sjit	Variation in time between packets sent from source.
Djit	Variation in time between packets sent to destination.
synack	Time between the SYN and the ACK packets.
ackdat	Time between the ACK packets.
$is\_sm\_ips\_ports$	Checks if source and destination IP and port are same.
ct_state_ttl	Count of connection states per destination TTL value.
$ct\_flw\_http\_mthd$	Count of HTTP methods seen in the traffic flow.
is_ftp_login	Indicates if an FTP login was successful.
ct_ftp_cmd	Number of FTP commands seen in the traffic.
ct_srv_src	Number of connections that have the same source and service.

Continued on next page

Table 3.3 – *Continued from previous page*

Feature	Simple Description
ct_srv_dst	Number of connections that have the same destination and service.
ct_dst_ltm	Number of connections to the same destination IP in the last minute.
ct_src_ltm	Number of connections from the same source IP in the last minute.
ct_src_dport_ltm	Number of connections with the same source IP and destination port in the last minute.
ct_dst_sport_ltm	Number of connections with the same destination IP and source port in the last minute.
ct_dst_src_ltm	Number of connections between the same source and destination IP in the last minute.
attack_cat	Type of attack (if any) identified in the traffic.
Label	Indicates whether the traffic is normal (0) or an attack (1).

Table 3.3: Descriptions of Features in UNSW-NB15 Dataset

3.3.1 Data Preprocessing

This section of the study paper carefully describes the steps that were taken to get the UNSW-NB15 dataset ready for in-depth analysis and then model training. This process is very important because it makes sure that the data is clean and organised in a way that makes it easier to get accurate insights and strong model performance.

At first, when the information is loaded into a Pandas DataFrame, important settings are made to make working with the data later easier. Also, a certain Pandas setting is changed so that all DataFrame fields can be shown. This is an important change because it lets you see the whole set of data at once, making sure that no detail is missed during the preparation steps.

After these initial arrangements, the dataset goes through a very important cleaning step. In this step, certain fields are dynamically removed from the DataFrame based on arguments given at runtime on the command line. This part of the preprocessing allows you to different study needs or specific analysis needs, this part of the preprocessing lets you be flexible and make changes as needed. At the same time, rows with missing values are found and deleted. This step is critical because missing data can cause models to make biased or incorrect predictions, resulting in lower total analysis accuracy.

After preprocessing, feature engineering is the next big step. This is where the raw dataset is changed into a shape that is better for analysis and training models. The

first thing that is done in this step is the one-hot encoding of the dataset's categorical values. It is possible for categorical variables to be changed into a set of binary variables, where each variable represents a possible group. This is accomplished by creating a new column for each category and inserting a 1 or 0 (true or false) in each column, depending on whether the category is present. For machine learning algorithms to work well, they need to be able to handle categorical data and figure out how different categories affect the expected outcomes.

At the same time, the numbers in the file are going through a normalisation process. Min-max scaling is used to change the data range to a normal scale, typically 0 to 1. The goal of this step is to make sure that features with higher ranges don't have an unfair effect on the model's decisions. This can happen because many machine learning algorithms, like neural networks and distance-based algorithms, are sensitive to the size of the input data. For a more fair and accurate learning process, normalisation makes sure that each feature adds about the same amount to the final prediction.

The dataset becomes a clean, well-structured, and analytically useful resource after these data cleaning, feature engineering, and normalisation steps are carefully carried out. This strict method for data preparation not only makes the data better, but it also makes it possible to use machine learning models correctly and effectively, which is explained in more detail later in the paper.

3.3.2 Splitting Data

Using the UNSW-NB15 dataset, two separate training and testing subsets with 175,341 and 82,332 instances, respectively, were created for this study. The purpose of this division is to guarantee both, a strong training environment and a useful framework for evaluation. The testing dataset verifies the accuracy and generalizability of these models across unseen data, while the training dataset enables the creation of models capable of detecting a variety of network intrusions. The division of network attacks into various categories, including Generic, Exploits, Normal, and others, makes it easier to comprehend and model each unique threat type in detail. The study aims to improve the intrusion detection models' predictive performance and reliability by maintaining a comprehensive representation across both datasets. This will ensure that the models are well-equipped to recognise and counteract a variety of dynamic cyber threats.

3.3.3 Model Training and Optimisation

In section 3.3.2, the dataset was carefully split into training and testing subsets. Now, the training phase is being looked at using SVM. SVM was chosen because it has a good track record of dealing with high-dimensional spaces and difficult classification problems in many areas, such as network security.

Category	UNSW_NB15_Training-Set	UNSW_NB15_Testing-Set
Normal	56,000	37,000
Generic	40,000	18,871
Exploits	33,393	11,132
Fuzzers	18,184	6,062
DoS	12,264	4,089
Reconnaissance	10,491	3,496
Analysis	2,000	677
Backdoor	1,746	583
Shellcode	1,133	378
Worms	130	44
Total	175,341	82,332

TABLE 3.4: UNSW-NB15 dataset's class distribution

SVM works by finding the best hyperplane that effectively separates the classes in the dataset. To do this, support vectors, which are the spaces between the closest data points of any class, are made as big as possible. This feature makes SVM perfect for the complicated task of analysing network traffic, where telling the difference between normal and malicious activity is often hard.

First, use Scikit-learn's svm module to set up the SVM classifier and choose a Radial Basis Function (RBF) kernel for training. The RBF kernel was chosen because it is capable of handling nonlinear data distributions, which are common in network intrusion data [Ham et al., 2023]. Important parts of the SVM configuration are the parameters C, which controls the regularisation, and gamma, which changes how far a single training example can reach. If the C value is low, the decision surface is smoother, which makes it easier to use but may not fit well enough. If the C value is high, it tries to perfectly classify all training examples, which could lead to overfitting. Gamma also changes how far a single training example's effect spreads, with lower values indicating a wider reach and higher values indicating a closer effect. It is very important to change these parameters so that the model can accurately capture the complex patterns in the dataset without becoming too perfect.

The process of parameter tuning involves trying out a lot of different values for C and gamma to find the best one that balances model complexity and training accuracy. To ensure that the model is sensitive enough to detect threats and strong enough to avoid false alarms, this tuning is necessary to make the SVM fit the network intrusion data's specifics.

After the model is trained, the reserved testing set is used to give it a thorough test. This part is very important because it shows how well the model can adapt to new data, which is a key sign of how useful it is in the real world. Performance metrics like accuracy, precision, recall, and the F1-score are used to get a full picture of how well the model works.

To sum up, training the SVM model is a very careful process that fits with the main goal of this research, which is to create a reliable and effective system for finding network intrusions. In this step, the model is not only customised to the dynamics of the dataset, but it can also make accurate predictions, which makes a big difference in improving security measures. This process shows how scientifically sound the method is and lays the groundwork for further deployment and real-time validation of the model. This makes it more useful and applicable for protecting network environments.

3.4 Performance Analysis of Models

Unexpectedly low accuracy of 28.46% has been shown in Figure 3.4, in the SVM model built for the UNSW-NB15 dataset. This part investigates possible theories for this less than ideal performance, based on the details of the model's configuration, the features of the dataset, and the kind of data preprocessing methods used.

	A	B	C	D
1		F1	Accuracy	
2	Analysis	0	0	
3	Backdoor	0	0	
4	DoS	0.043509	0.022744	
5	Exploits	0.625687	0.833902	
6	Fuzzers	0.075983	0.104586	
7	Generic	0.977229	0.956282	
8	Normal	0.70713	0.663297	
9	Reconnais	0.311037	0.266018	
10	Shellcode	0	0	
11	Worms	0	0	
12	Macro_Av	0.274057	0.284683	
13				

FIGURE 3.4: Description of the Image

3.4.1 Inadequate Model Parameters

The selection of model parameters is one of the main aspects maybe causing the SVM model to perform poorly. With parameters gamma=0.5 and C=0.1, the model makes use of an RBF kernel. An overfitting situation where the model is overly sensitive to the noise in the training data may result from a too high gamma parameter, which determines the influence of individual training examples. On the other hand, a too low C parameter can cause an underfitting scenario in which the model oversimplifies the

decision boundary and compromises between obtaining a low error on the training data and preserving a smooth decision boundary.

3.4.2 Issues with Data Preprocessing

Effectiveness of machine learning models is mostly dependent on data preprocessing. Here, preprocessing is encoding categorical variables with OneHotEncoder and scaling numerical features with MinMaxScaler. Should these methods be used improperly, the underlying connections in the data may be distorted. Additionally, the script lets you exclude features according to input arguments, which could unintentionally eliminate important predictors and make it harder for the model to generalise from the training data.

3.4.3 Complexity of the Dataset

Targeted at network intrusion detection, the UNSW-NB15 dataset contains by nature complex and subtle patterns that might not be easily visible through simple non-linear margins or linearly separable. Even with a non-linear kernel, the SVM model's simplicity may not be enough to really capture these complex patterns.

3.4.4 Evaluation Metric Considerations

In a dataset probably characterised by class imbalances and several attack types, depending only on accuracy as the performance metric might not give a complete picture of the model's efficacy. Precision and recall are some metrics that might provide more detailed information about how well the model performs in various classes.

3.4.5 Recommendations for Development

The performance of the model should be improved by a number of modifications. First of all, finding better values for C and gamma may be made easier by using a methodical method of parameter tuning, such grid search with cross-validation. Second, it is essential to review the preprocessing pipeline to guarantee best feature encoding and scaling. Thirdly, because they are better able to manage high-dimensional data and difficult pattern recognition tasks, advanced feature selection methods and investigating other machine learning models like Random Forests or Neural Networks may produce better performance.

In conclusion, a combination of perhaps inadequate parameter settings, inadequate data preprocessing, and the difficult nature of the dataset can be blamed for the SVM model's poor performance on the UNSW-NB15 data. Full resolution of these problems

3.5 Feature Analysis

This section evaluates the specific impact of each feature on the accuracy of the model. A direct approach is utilised: one feature is eliminated from the dataset, which is subsequently inputted into the SVM model. This process is iterated for each column in the dataset, guaranteeing a comprehensive evaluation of the influence of each feature on the model's performance. To achieve this, multiple scripts are used.

3.5.1 Automation

To fully understand how different features affect the effectiveness of the SVM model used for finding security threats in MANETs, a thorough automated script was created. This script reruns the SVM model by removing each feature from the dataset in a planned way, this script reruns the SVM model to see how the lack of each feature affects the accuracy of the main model. This methodical approach is used to accurately assess the importance of each feature in the model.

The process starts with a Python script that inspects the directory for a certain dataset in CSV format that was given as an argument on the command line. After finding the file, the script loads the data into a Pandas DataFrame. Later, it extracts the names of all features except the target label $attack_cat$ and saves them in a text file called "Features.txt." Starting with this step is very important for laying the groundwork for later feature elimination trials.

Iterates through each feature listed in "Features.txt" until all of the names are written down. Within each cycle, the script starts a second Python script called "SVM.py," which is specifically made to run the SVM model without the feature being looked at at the moment. After the model is trained on the new dataset, this secondary script checks how well it did by removing the chosen feature. The results of these tests, which include key performance metrics like loss and accuracy, are saved as separate CSV files for each feature that was tested. This creates a complete set of data files that show how the model's performance decreased or increased when certain features were removed.

In this automated, iterative process, each feature is processed separately, showing how each one helps the system detect things. It usually takes a little more than an hour for the script to run all 45 features and compile 45 CSV files. By looking at the performance

impact of each feature, you can see exactly which features are necessary to maintain accuracy and which ones may be unnecessary or less important.

For improving the SVM model, this feature impact assessment is very helpful. Model performance can be improved while making it easier to understand by figuring out the most important factors that have a big impact on accuracy. This increases the model's usefulness in real-life situations where computer power is limited, as well as making it work better. The investigation into this problem helps shape the ongoing development and tuning of the intrusion detection system, making sure it stays strong against changing security threats in MANET settings.

3.5.2 Data Visualisation

This part talks about a Python script that lets you see how different features affect a model's F1-Score, which is an important way to judge how accurate the model is (elaborated in section 4.0.2). This script uses Pandas to change the data in several CSV files that represent different model scenarios, as well as Matplotlib to create visual outputs. The main focus is on how taking away each feature changes the performance compared to a model where no features are taken away (the baseline model).

The process loads each CSV file from the "reports" directory using a Pandas DataFrame. The model's performance metrics are recorded in each file in a different configuration. One file is the baseline, which has all features, while the others have one feature removed at a time. The filenames without the extensions are used as keys in a dictionary called dfs, which makes it easy to organise the data for further analysis.

As the script goes through each DataFrame in dfs, it gets the F1-Score for each configuration from the "accuracy" row. These scores are very important for figuring out what will happen when each feature is taken away. The datasets' names are used to find the scores in a dictionary called f1s. This allows us to easily determine the difference in F1-Score between each dataset and the baseline.

The differences in the dictionary f1_diffs show how each feature changed the F1-Score of the model. In this case, a negative value means that removing the feature makes the model perform much worse, showing how important it is to the model. A positive value, on the other hand, means that taking away the feature doesn't hurt the model's performance and might even slightly improve it. This means that the feature may not be needed or isn't very important.

A horizontal bar chart displaying these differences makes the information clear. The length and color of each bar indicate how big and what kind of effect each feature has. Labels with numbers on each bar show the exact values, which makes the graph easier to understand. The chart is called "Features to F1-Score Contribution Comparison," and

it has two detailed axes called "Features" and "Contribution to F1-Score." The x-axis range was chosen to draw attention to these differences, and the vertical line at x = 0 makes it easy to see which features have positive and negative effects.

Chapter 4

Results

This section shows the findings of the SVM Model run.

4.0.1 Model Accuracy

In this project, significant insights were gained into how certain network features within MANETs contribute to the identification of malicious nodes. A detailed graph can be see Fig 4.1. A pivotal feature, 'is_same_ips_ports', which checks if source and destination IP addresses and ports match, is crucial for detecting spoofing or reflection attacks. These attacks are common in MANET environments where attackers might impersonate legitimate nodes. Additionally, 'stcpb', representing the source to destination TCP base sequence number, and 'sinpkt', which measures the inter-arrival time between packets from the source to the destination, are essential for identifying abnormal behaviors indicative of network probing and denial-of-service attacks. These attacks often involve unusual packet timing and sequences that disrupt normal network operations. The correct understanding and application of these features are paramount in enhancing the model's accuracy and the network's resilience against the diverse threats present in dynamic network environments like MANETs. A structured

Feature	**Description**	**Reduction in F1 Score**
is_same_ips_ports	Checks if source and destination IP and port are same.	-0.0024
stcpb	TCP sequence number of the first packet from source.	-0.0084
sinpkt	Measures the time interval between packets sent from the source.	-0.0003
dtcpb	TCP sequence number of the first packet to destination.	-0.0090

Table 4.1: Impact of Feature Removal on F1 Score

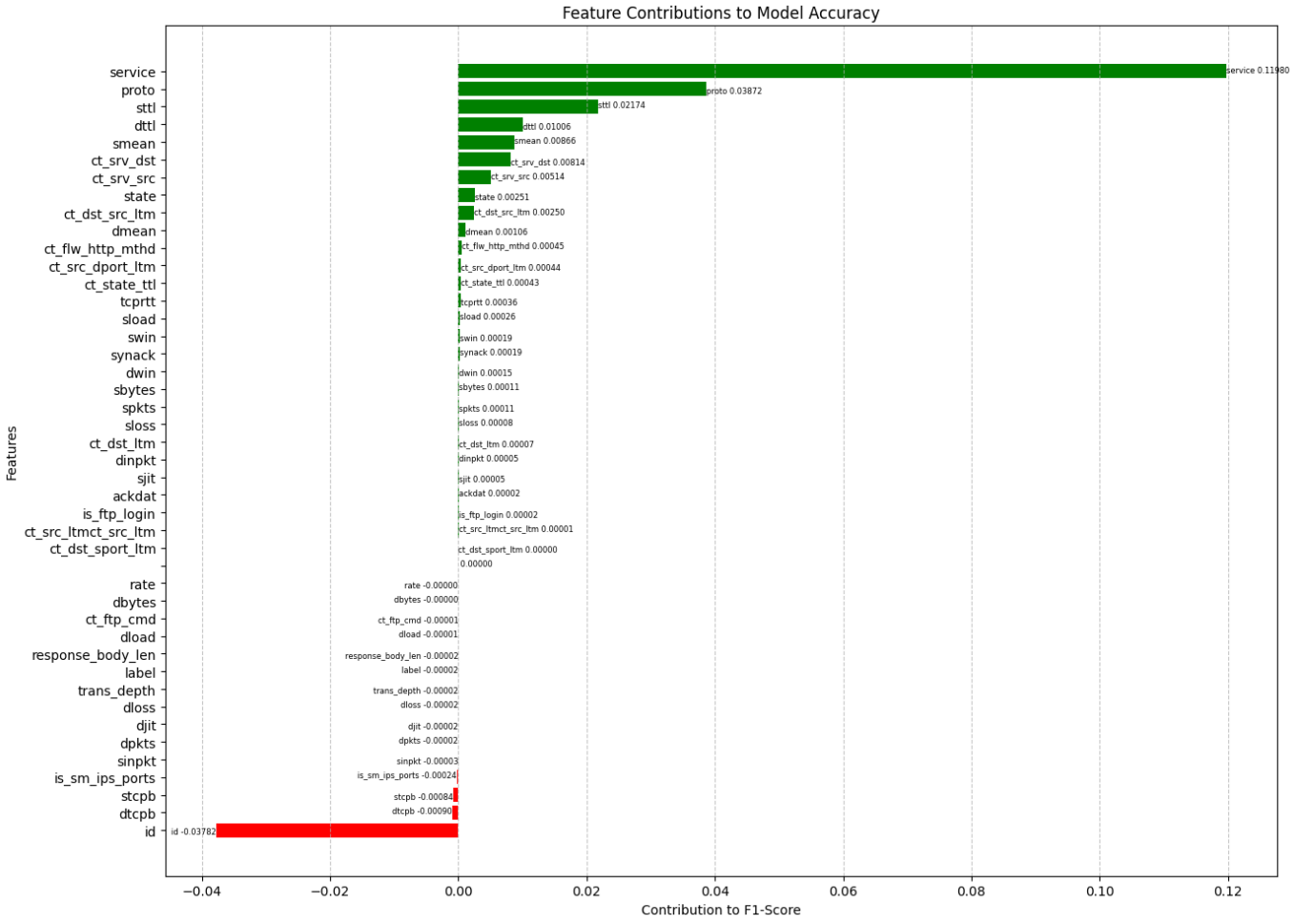

FIGURE 4.1: Differential Impact of Feature Omission on Model's F1-Score

4.0.2 Evaluation Metrics

Two Main Metrics, Accuracy and F1 Score,were used to evaluate the performance of the classification models. In order to give a thorough picture of the model's performance in a variety of scenarios, these metrics were selected.

Accuracy: This metric measures the proportion of true results (both true positives and true negatives) among the total number of cases examined.

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

Where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives.

F1 Score: The $F1$ Score is the harmonic mean of Precision and Recall, providing a balance between the two when the dataset is imbalanced. It is particularly useful when the costs of false positives and false negatives are very different.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where:

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

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

Chapter 5

Conclusions

This study thoroughly investigated the use of machine learning methods, specifically Support Vector Machines (SVM), to improve intrusion detection in Mobile Ad-Hoc Networks (MANETs), utilising the comprehensive UNSW-NB15 dataset. The analysis emphasised the crucial importance of selecting relevant features and fine-tuning the model in order to optimise the accuracy of detection. By conducting simulations in Network Simulator 3 (NS3), we gained a detailed understanding of the dynamics of Mobile Ad hoc Networks (MANETs), which allowed us to develop a customised approach to training Support Vector Machine (SVM) models. The simulations offered an authentic framework to assess the SVM's capacity to distinguish between benign and malicious activities in diverse network conditions. The results obtained from these simulations played a crucial role in adjusting the SVM parameters, such as the kernel type, gamma, and C, to make sure that the model was strong enough to handle the various types of attack vectors found in the dataset, including DoS, Exploits, and Generic attacks. This study significantly enhances the cybersecurity field by improving the ability to detect sophisticated network threats, which is crucial for protecting against them. As a result, it benefits both academic research and practical applications in network security.

Chapter 6

Project Management

6.1 Project Challanges

A significant challenge in this project is the limited availability of resources and information online, which made research and development difficult. This scarcity is partly due to the specialized nature of MANETs and the rapid pace of technological advancements in this field.

The first challenge faced was in researching about NS3. Once the network was developed, it was extremely complicated and time consuming to extract the features using trace file in NS3. Utilisation of PCAP files was also considered and tested to extracting features, however due to lack of online support meant that this entire implementation would need to be manually designed. The steep learning curve required for this task would have taken an immeasurable amount of time. Since the initial goal of the projects required further work such as dataset cleansing, a complete SVM model creation, and time to train the model and analyse the model, this complex application of pcap files could not be pursued.

Following the usage of UNSW-NB15 dataset, it was found that it contained redundant information that needed to be cleaned manually. A SVM model was then made based on its structure and while splitting the dataset into 30-70, it was hypothesised that the classifiers were equally divided within that dataset which led to in imbalance and subsequent incorrect results near the end of the project.

To address the challenges encountered, a new SVM model was promptly developed and executed overnight. The model generated 13 out of the required 45 models in a four-hour span. This effort extended the total simulation time to over 12 hours, nearing the project deadline. Nevertheless, these obstacles were successfully overcome, and the results were thoroughly analyzed and documented in this report.

This project necessitated a steep learning curve, as there was limited guidance available from sources such as previous research reports and documentation on development applications. Due to this, it offered a realistic experience for conducting similar industry research, paving the way for a more progressive approach in future projects.

6.2 Gantt Chart

The following two Gantt charts show the predicted and actual progression throughout the academic year. It can be seen in the Actual Gantt Chart 6.2 that before December, no progress is visible for this project, this is due to the fact that the project scope changed. The previous project was discarded and a fresh project was initiated. This contributed to the constricted time frame for the presented implementation.

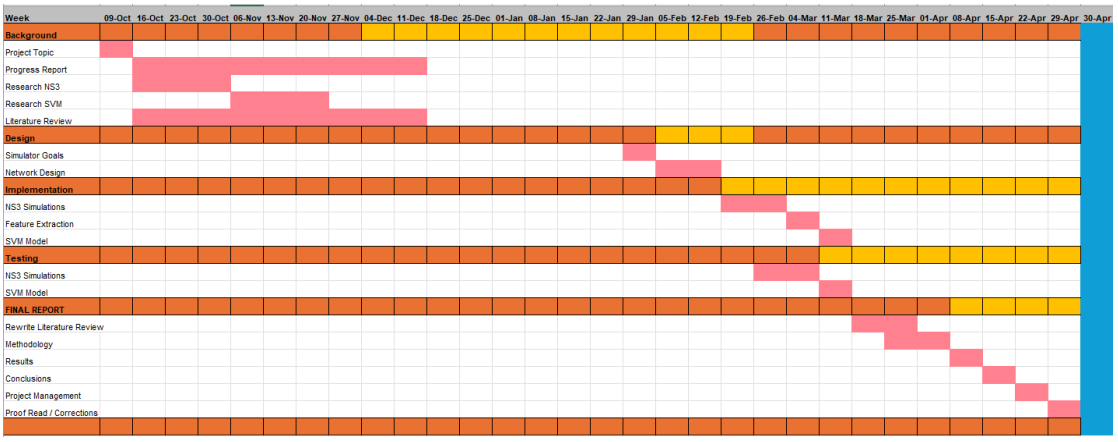

FIGURE 6.1: Predicted Progress Gantt chart

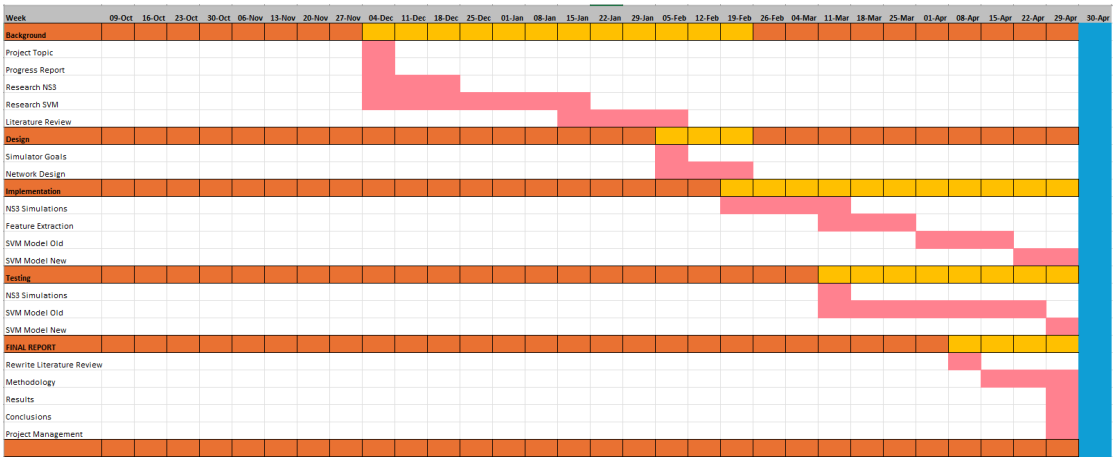

FIGURE 6.2: Actual Progression Gantt chart

6.3 Risk Assessment

The following risk assessment was conducted on aspects of this project:

TABLE 6.1: Risk Assessment Table

Problem	**Severity (1-5)**	**Prob (1-5)**	**Risk (1-5)**	**Mitigation Strategy**
Loosing project files	4	2	8	Make backups of the files, use version control
Implementation takes too long	3	3	9	Work on the project daily and meet the weekly goals
Learning C++ takes longer than expected	4	2	8	Use online tutorials and university resources (e.g., programming support desk) for help
Learning to use NS3	4	4	16	Use prebuilt PCAP files to train data
Learning about SVM Technique	2	2	4	Learn through online tutorials, in case of doubts refer someone senior
The final result doesn't meet project aims	4	3	12	Reflect on project aims throughout
Unable to meet weekly deadlines due to other modules	4	3	12	Delegate time for project and other modules

Bibliography

Khattab M.Ali Alheeti, Anna Gruebler, and Klaus McDonald-Maier. Intelligent Intrusion Detection of Grey Hole and Rushing Attacks in Self-Driving Vehicular Networks. *Computers 2016, Vol. 5, Page 16*, 5(3):16, 7 2016a. ISSN 2073-431X. doi: 10.3390/COMPUTERS5030016.

Khattab M.Ali Alheeti, Anna Gruebler, and Klaus D. McDonald-Maier. An Intrusion Detection System against Black Hole Attacks on the Communication Network of Self-Driving Cars. *Proceedings - 2015 6th International Conference on Emerging Security Technologies, EST 2015*, pages 86–91, 3 2016b. doi: 10.1109/EST.2015.10.

Ankita Chourasia and Sanjiv Tokekar. Reinforcement Learning based Security Policy to Mitigate Wormhole, Blackhole and Grayhole Attacks in MANET. *2024 2nd International Conference on Computer, Communication and Control, IC4 2024*, 2024. doi: 10.1109/IC457434.2024.10486553.

Sihem Dadi and Mohamed Abid. Enhanced Intrusion Detection System Based on AutoEncoder Network and Support Vector Machine. *Smart Innovation, Systems and Technologies*, 237:327–341, 2022. ISSN 2190-3026. doi: 10.1007/978-981-16-3637-0{_}23.

Aditya Gupta, Prabal Verma, and Rakesh Singh Sambyal. An Overview of MANET: Features, Challenges and Applications. *NCRACIT) International Journal of Scientific Research in Computer Science, Engineering and Information Technology © 2018 IJSRCSEIT*, 1(4):2456–3307, 2018.

Seung Mi Ham, Hye Min Lee, Jae Hyun Lim, and Jeongwook Seo. A Negative Emotion Recognition System with Internet of Things-Based Multimodal Biosignal Data. *Electronics 2023, Vol. 12, Page 4321*, 12(20):4321, 10 2023. ISSN 2079-9292. doi: 10.3390/ELECTRONICS12204321. URL https://www.mdpi.com/2079-9292/12/20/4321/htmhttps://www.mdpi.com/2079-9292/12/20/4321.

Charles Hutchins, Leonardo Aniello, Enrico Gerding, and Basel Halak. MANET-Rank: A Game Theoretic Evaluation Framework for Defence Protocols against Packet Dropping Attacks in MANETs. Network Operations and Management Symposium, 2024.

Luca Divit. lucadivit/Pcap_Features_Extraction: This program allow you to extract some features from pcap files. URL https://github.com/lucadivit/Pcap_Features_Extraction.

Nour Moustafa and Jill Slay. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *2015 Military Communications and Information Systems Conference, MilCIS 2015 - Proceedings*, 12 2015. doi: 10.1109/MILCIS.2015.7348942.

Nour Moustafa and Jill Slay. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective*, 25(1-3):18–31, 4 2016. ISSN 19393547. doi: 10.1080/19393555.2015.1125974.

Nour Moustafa, Gideon Creech, and Jill Slay. Big Data Analytics for Intrusion Detection System: Statistical Decision-Making Using Finite Dirichlet Mixture Models. pages 127–156, 2017. ISSN 2520-1867. doi: 10.1007/978-3-319-59439-2{_}5.

Nour Moustafa, Jill Slay, and Gideon Creech. Novel Geometric Area Analysis Technique for Anomaly Detection Using Trapezoidal Area Estimation on Large-Scale Networks. *IEEE Transactions on Big Data*, 5(4):481–494, 12 2019. ISSN 23327790. doi: 10.1109/TBDATA.2017.2715166.

Charles E. Perkins and Elizabeth M. Royer. Ad-hoc on-demand distance vector routing. *Proceedings - WMCSA'99: 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, 1999. doi: 10.1109/MCSA.1999.749281.

T. Poongothai and K. Duraiswamy. Intrusion detection in mobile AdHoc networks using machine learning approach. *2014 International Conference on Information Communication and Embedded Systems, ICICES 2014*, 2 2015. doi: 10.1109/ICICES.2014.7033949.

Mohanad Sarhan, Siamak Layeghy, Nour Moustafa, and Marius Portmann. NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 371 LNICST:117–135, 2021. ISSN 1867822X. doi: 10.1007/978-3-030-72802-1{_}9/TABLES/13.

Sandeep Singh, Anshu Bhasin, and Anshul Kalia. Capitulation of mitigation techniques of packet drop attacks in MANET to foreground nuances and ascertain trends. *International Journal of Communication Systems*, 34(10):e4822, 7 2021. ISSN 1099-1131. doi: 10.1002/DAC.4822.

N. Sivanesan and K. S. Archana. Performance Analysis of Machine Learning-based Detection of Sinkhole Network Layer Attack in MANET. *International Journal of*

Advanced Computer Science and Applications, 13(12):512–520, 34 2022. ISSN 2156-5570. doi: 10.14569/IJACSA.2022.0131262.

N. Sivanesan, A. Rajesh, S. Anitha, and K. S. Archana. Detecting Distributed Denial of Service (DDoS) in MANET Using Ad Hoc On-Demand Distance Vector (AODV) with Extra Tree Classifier (ETC). *Iranian Journal of Science and Technology - Transactions of Electrical Engineering*, pages 1–15, 12 2023. ISSN 23641827. doi: 10.1007/S40998-023-00678-7/FIGURES/12.

The ns-3 Network Simulator Project. ns-3 — a discrete-event network simulator for internet systems. URL https://www.nsnam.org/.

Jingjing Wang and Chunxiao Jiang. Flying Ad Hoc Networks. 2022. doi: 10.1007/978-981-16-8850-8.

Zulfiqar Ali Zardari, Jingsha He, Nafei Zhu, Khalid Hussain Mohammadani, Muhammad Salman Pathan, Muhammad Iftikhar Hussain, and Muhammad Qasim Memon. A Dual Attack Detection Technique to Identify Black and Gray Hole Attacks Using an Intrusion Detection System and a Connected Dominating Set in MANETs. *Future Internet 2019, Vol. 11, Page 61*, 11(3):61, 3 2019. ISSN 1999-5903. doi: 10.3390/FI11030061.