

Intrusion Detection and Classification in Mobile Ad Hoc Networks Using Machine Learning



Lemessa Aschala Temesgen

A Thesis Submitted to the Department of Computer Science and
Engineering

College of Electrical Engineering and Computing

Presented in Partial Fulfilment of the Requirement for the Degree of
Master's in Computer Science and Engineering (Specialization in
Networking and IoT)

Office of Graduate Studies

Adama Science and Technology University

February, 2025

Adama, Ethiopia

Intrusion Detection and Classification in Mobile Ad Hoc Networks Using Machine Learning

Lemessa Aschala Temesgen

Advisor: Dr. Ketema Adere

A Thesis Submitted to the Department of Computer Science and
Engineering

College of Electrical Engineering and Computing

Presented in Partial Fulfilment of the Requirement for the Degree of
Master's in Computer Science and Engineering (Specialization in
Networking and IoT

Office of Graduate Studies

Adama Science and Technology University

February, 2025

Adama, Ethiopia

DECLARATION

I hereby declare that this Master's thesis entitled “*Intrusion Detection and Classification in Mobile Ad Hoc Networks Using Machine Learning*” is my original work. That is, it has not been submitted for the award of any academic degree, diploma, or certificate in any other university. All the sources of materials that are used for this thesis have been duly acknowledged through citation.

Lemessa Aschala Temesgen

Name of Student

Signature

Date

RECOMMENDATION OF ADVISORS

I, the major advisor of this thesis, hereby certify that I have read the revised version of the thesis “*Intrusion Detection and Classification in Mobile Ad Hoc Networks Using Machine Learning*” prepared under our guidance by Lemessa Aschala Temesgen submitted in partial fulfilment of the requirements for the degree of Masters of Science in Computer Science and Engineering. Therefore, I recommend submitting the revised version of the thesis to the department following the applicable procedures.

_____	_____	_____
Major Advisor	Signature	Date
_____	_____	_____
Co-advisor	Signature	Date

APPROVAL PAGE

I, the advisor of the thesis entitled “*Intrusion Detection and Classification in Mobile Ad Hoc Networks Using Machine Learning*” developed by Lemessa Aschala Temesgen, hereby certify that the recommendation and suggestion made by the board of examiners are appropriately incorporated into the final version of the thesis.

_____ Major Advisor	_____ Signature	_____ Date
------------------------	--------------------	---------------

_____ Co-advisor	_____ Signature	_____ Date
---------------------	--------------------	---------------

We, the undersigned, members of the Board of Examiners of the thesis by Lemessa Aschala Temesgen, have read and evaluated the thesis entitled “*Intrusion Detection and Classification in Mobile Ad Hoc Networks Using Machine Learning*” and examined the candidate during open defence. This is, therefore, to certify that the thesis is accepted for partial fulfilment of the requirement of the Master of Science in Computer Science and Engineering degree.

_____ Chairperson	_____ Signature	_____ Date
----------------------	--------------------	---------------

_____ Internal Examiner	_____ Signature	_____ Date
----------------------------	--------------------	---------------

_____ External Examiner	_____ Signature	_____ Date
----------------------------	--------------------	---------------

The thesis's final approval and acceptance are contingent upon the submission of its final copy to the Office of Postgraduate Studies (OPGS) through the Department Graduate Council (DGC) and School Graduate Committee (SGC).

_____ Department Head	_____ Signature	_____ Date
--------------------------	--------------------	---------------

_____ School Dean	_____ Signature	_____ Date
----------------------	--------------------	---------------

_____ Office of Postgraduate Studies, Dean	_____ Signature	_____ Date
---	--------------------	---------------

ACKNOWLEDGMENT

I am pleased to express my deepest gratitude to Dr. Ketema Adere, who is very kind and generous and has devoted much of his time and expertise to helping, guiding, and advising me throughout this research work. Indeed, this work results from his continuous efforts and broad experience in Computer Engineering. He has been a constant presence throughout my study and has kept me motivated, challenged, and productive. I would also like to extend my heartfelt acknowledgments to the College of Electrical and Computing and Computer Science and Engineering Department, which provided valuable laboratory experiments during this research work. Lastly, I would like to thank my family and friends for their love and moral support through their prayers. I am deeply grateful to all of them. I am pent-up to the highest degree indebted to the Almighty God, who gave me health and the ability to reach here with happiness and glory.

TABLE OF CONTENTS

DECLARATION	i
RECOMMENDATION OF ADVISORS	ii
APPROVAL PAGE	iii
ACKNOWLEDGMENT	iv
TABLE OF CONTENTS	v
LIST OF FIGURES	x
LIST OF TABLES	xi
LIST OF ABBREVIATIONS.....	xii
ABSTRACT	xiii
CHAPTER ONE	1
INTRODUCTION	1
1.1 Background of the Study.....	1
1.2 Motivation of the Study.....	3
1.3 Statement of Problem.....	4
1.4 Research questions.....	5
1.5 Objectives.....	5
1.5.1 General Objective of the Study	5
1.5.2 Specific Objectives of the Study.....	5
1.6 Significance of the Study	5
1.7 The Scope of the Study	6
1.8 Limitations of the Study.....	6
1.9 Organization of the Paper.....	7
CHAPTER TWO	8
LITERATURE REVIEW AND RELATED WORKS	8

2.1	Introduction	8
2.2	AD Hoc Network	8
2.3	Applications of MANET	10
2.3.1	Military Communications	10
2.3.2	Emergency and Disaster Response	10
2.3.3	Search and Rescue Operations	11
2.3.4	Vehicular Ad-hoc Networks (VANETs).....	11
2.3.5	Crowd Sensing and Monitoring.....	11
2.3.6	Smart Agriculture	11
2.3.7	Sensor Networks in Remote Areas	12
2.3.8	Education and Training	12
2.3.9	Social Networking	12
2.3.10	IoT Communication	12
2.4	Protocol Layers of MANET	13
2.4.1	Physical Layer	13
2.4.2	Data Link Layer	14
2.4.3	Network Layer and Routing Protocol	16
2.4.3.1	Routing Management	16
2.4.3.2	Topology-Based Routing.....	17
2.4.3.3	Position-Based Routing	19
2.4.3.4	Energy Aware Routing	19
2.4.4	Transport Layer	20
2.4.5	Application Layer	21
2.5	Security Attacks in MANET	21
2.6	An intrusion detection system	23

2.6.1	Intrusion Detection and Classification System in MANET	24
2.6.2	Intrusion Detection and Classification Models in the MANET	27
2.7	Machine Learning Model	28
2.7.1	Supervised Learning	28
2.7.2	Unsupervised Learning	28
2.7.3	Semi-Supervised Learning	29
2.7.4	Reinforcement Learning.....	29
2.7.5	Deep Learning	29
2.8	Dataset.....	29
2.9	Related Works	31
2.10	Summary.....	36
CHAPTER THREE.....		37
MATERIALS AND METHODOLOGY		37
3.1	Introduction	37
3.2	Research Design	37
3.3	Research Framework.....	37
3.4	Data Pre-processing	38
3.4.1	Feature Normalization.....	39
3.4.2	Feature Selection.....	39
3.4.3	Hyperparameters Values	40
3.5	Proposed Machine Learning Models for Intrusion Detection	40
3.5.1	Decision Tree Model.....	40
3.5.2	Random Forest Model.....	42
3.5.3	Support Vector Machine Model	43
3.5.4	K-nearest Neighbours Classification (KNN)	44

3.5.5	Ensemble Method	45
3.6	Performance Evaluation Metrics	46
CHAPTER FOUR.....		47
PROPOSED MODEL AND IMPLEMENTATION.....		47
4.1.	Proposed Model Architecture and Design.....	47
4.2	Description of Block Diagram	47
4.3	Implementation of the Proposed Model	49
4.3.1	Dataset Preparation.....	49
4.3.2	Dataset Feature Extraction.....	50
4.3.3	Experimental Setup	51
4.3.3.1	K-Nearest Neighbours Training.....	52
4.3.3.2	Decision Tree Model Training	53
4.3.3.3	Random Forest Model Training	53
4.3.3.4	Support Vector Machine Model Training.....	54
4.3.3.5	Ensemble Model Training	55
CHAPTER FIVE		56
RESULTS AND DISCUSSIONS.....		56
5.1	Introduction	56
5.2	. Model Evaluation and Performance Analysis	56
5.2.1	Evaluation of K-nearest Neighbours Model.....	56
5.2.2	Evaluation of Decision Tree Model.....	58
5.2.3	Evaluation of Random Forest Model.....	59
5.2.4	Evaluation of the SVM Model.....	60
5.2.5	Evaluation of Ensemble Model	61
5.3	Comparative Analysis with Recent Studies	64

5.4	Summary of Findings.....	64
5.5	Discussion.....	66
CHAPTER SIX.....		69
CONCLUSION AND RECOMMENDATIONS		69
6.1.	Conclusion.....	69
6.2.	Contributions of the Thesis	70
6.3.	Recommendations for Future Work.....	70
REFERENCES		72

LIST OF FIGURES

Figure 2.1: TCP/IP and OSI model	13
Figure 2.2: IEEE 802.11 frame format in bytes.....	14
Figure 2.3: IEEE 802.11 Frame control field in bits.....	14
Figure 2.4: Transmission with RTS/ CTS	15
Figure 2.5: Transmission without RTS/ CTS	15
Figure 3.1: Research Framework.....	38
Figure 4.1: Proposed Intrusion Detection and Classification Approach.....	47
Figure 4.2: Staking Ensemble Model.....	55
Figure 5.1: Classification Report and Confusion Matrix of KNN.....	57
Figure 5.2: Classification Report and Confusion Matrix of Decision Tree Model	58
Figure 5.3: Classification Report and Confusion Matrix of Random Forest Model	59
Figure 5.4: Classification Report and Confusion Matrix of SVM Model.....	61
Figure 5.5: Classification Report and Confusion Matrix of Ensemble Model	62
Figure 5.6: Receiver Operating Characteristic (ROC) curve for the dataset.....	63

LIST OF TABLES

Table 2.1: Summary Table of Attack Types in UNSW-NB15.....	23
Table 2.2: Summary of the Related Work.....	32
Table 4.1: UNSW-NB15 attack distribution	50
Table 4.2: Dataset description of UNSW-NB15 dataset.....	50
Table 5.1: Dataset with features selected by RF-RFE (16 features).....	62

LIST OF ABBREVIATIONS

AR	Attribute Ratio
CART	Classification and Regression Tree
CIA	Confidentiality, Integrity, And Availability
DDoS	Distributed Denial of Service
DIDS	Distributed Intrusion Detection System
DOS	Denial of Service
DT	Decision Tree
FAR	False Alarm Rate
FN	False Negative
FP	False Positive
IDS	Intrusion Detection System
KNN	K-Nearest Neighbors
MANET	Mobile Adhoc Network
ML	Machine Learning
NIDS	Network-Based Intrusion Detection System
PCA P	Principal Component Analysis
RF	Random Forest
RF-RFE	Random Forest - Recursive Feature Elimination
SVM	Support Vector Machine
t-SNE	T-Distributed Stochastic Neighbor Embedding
VANET	Vehicular Adhoc Network

ABSTRACT

Mobile Ad Hoc Networks (MANETs) are decentralized, self-organizing networks that operate without fixed infrastructure, making them highly flexible and vulnerable to various security threats. Ensuring security in MANETs is challenging due to their dynamic topology, lack of centralized monitoring, and resource constraints. Traditional security mechanisms, such as encryption and authentication, often fail to detect sophisticated intrusion attacks in these networks. Existing intrusion detection systems (IDS) struggle with high false alarm rates, adaptability to evolving attacks, and computational overhead, creating a significant research gap in MANET security. Motivated by the increasing security threats and the inadequacy of existing methods, this study aims to develop an adaptive and efficient machine learning-based intrusion detection and classification system for MANETs. The research investigates the effectiveness of multiple machine learning classifiers, including Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), and an ensemble-based approach, in detecting and classifying network intrusions. The study utilizes the UNSW-NB15 dataset, which contains diverse attack types, and employs feature selection techniques to optimize model performance while minimizing computational costs. The proposed ensemble model integrates multiple classifiers to improve detection accuracy and robustness. Experimental results demonstrate that the ensemble model outperforms individual classifiers, achieving an accuracy of 98.53%, a precision of 98.79%, a recall of 98.78%, and an F1-score of 98.78%. This improvement significantly reduces false positives while maintaining high detection capability, addressing key limitations of traditional IDS. This study's contributions include the development of an efficient, adaptive ML-based IDS tailored for MANETs, a comparative analysis of machine learning models for intrusion detection, and the application of feature selection to enhance detection efficiency. The findings provide valuable insights for securing MANET environments and lay the foundation for future real-time adaptive intrusion detection research.

Keywords: Machine Learning, Denial of Service, Detection and Classification, Mobile Ad Hoc Networks, Ensemble-Based Detection, UNSW-NB15 Dataset, Network Security.

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

Mobile Ad Hoc Networks (MANETs) are decentralized, self-configuring wireless networks composed of mobile nodes communicating without relying on fixed infrastructure. These networks are extensively used in military operations, disaster response, vehicular networks, and remote communication applications. MANETs face significant security challenges despite their flexibility due to their open medium, lack of centralized control, dynamic topology, and resource constraints. Intrusions such as denial-of-service (DoS) attacks, black hole attacks, Sybil attacks, and wormhole attacks are prevalent, posing serious threats to data integrity, availability, and confidentiality. In Ethiopia and across Africa, MANETs are increasingly utilized in defense, telecommunications, and emergency response sectors. However, cyber threats have escalated, with a notable rise in intrusion attempts targeting critical infrastructures. Reports from INSA (Information Network Security Administration) indicate a growing number of cyberattacks on Ethiopia's telecommunications and defense networks, with over 5,000 cyber incidents recorded in 2023 alone. Ethio Telecom and the Ministry of Defense (MoD) have also acknowledged vulnerabilities in their mobile and tactical communication networks, necessitating improved security mechanisms.

A lack of robust intrusion detection systems (IDS) has made it difficult to detect and mitigate these threats effectively, exposing MANETs to potential exploitation. MANETs are highly flexible and can be deployed in various scenarios where establishing a fixed infrastructure is impractical or impossible, such as military operations, emergency response situations, and remote areas with limited connectivity. However, MANETs face numerous security challenges due to their unique characteristics. Unlike traditional networks, MANETs lack a centralized authority and operate dynamically and self-organisingly (Yi et al., 2001). Wireless networks are susceptible to attacks due to their nature, requiring every node to be prepared for defense. Additionally, packet losses in MANET can result from node-related issues, congestion, or mobility factors (Tilwari et al., 2020). In MANETs, packet losses can occur due to node-related issues (selfish or malicious behaviour), congestion (queuing problems, busy channels, link interference), and mobility factors (MAC and routing layer issues).

Decentralized decision-making and reliance on cooperative algorithms make mobile computing environments vulnerable to attacks targeting these weaknesses. While encryption and authentication help reduce intrusions in MANETs, they cannot eliminate them. Unlike wired networks, MANETs lack centralized infrastructure, posing challenges for intrusion identification systems. Machine learning techniques offer a promising approach to efficiently classify malicious and normal nodes in MANETs (Sivanesh & Sarma Dhulipala, 2022), (Arden et al., 2021). Wireless nodes are susceptible to DoS attacks, especially DDoS attacks from multiple sources, aiming to disrupt legitimate service usage by overwhelming resources. Despite extensive research, existing remedies have not effectively countered DoS attacks in practical MANET scenarios (Islam et al., 2021). MANETs are popular due to their decentralized nature, but they face challenges like open network architecture and dynamic topology, making them vulnerable to DoS attacks. Security measures like firewalls and filtering are used, but they have limitations regarding distinguishing attacks and resource constraints. Traceback helps identify attackers, but it is hindered by mobility. Pushback addresses congestion but is ineffective in specific deployments (Virupakshar et al., 2020). Traditional security mechanisms, such as encryption, authentication, and intrusion detection systems, are unsuited for MANETs. MANETs have resource-constrained devices with limited processing power, memory, and energy. Moreover, the frequent topology changes in MANETs can result in high communication overhead and latency, challenging deploying traditional security mechanisms. The lack of a centralized monitoring point makes detecting and mitigating attacks difficult (Biswas et al., 2021). Current research explores traditional machine learning algorithms to mitigate DoS and DDoS attacks in MANETs. Authors have presented methods like KNN with correlation analysis and SVM for detection. Additionally, they have compared algorithms like SVM, ANN, NNC, GBC, and NBC for detecting P2P bots. By dividing data into training and testing sets, machine learning models offer improved performance on novel attacks, addressing the limitations of traditional methods (Abubakar et al., 2020), (Alauthman et al., 2020).

Machine learning algorithms can analyze large volumes of network traffic data, identify patterns, and detect anomalies associated with malicious activities. By leveraging machine learning, it is possible to develop intelligent systems that can adapt to evolving attack patterns and detect attacks in real-time. The application of machine learning in MANET security can provide several benefits. First, it can enable the detection of previously unknown and emerging attack patterns

that traditional rule-based or signature-based approaches may not capture. Second, machine learning models can learn from historical network data, allowing for proactive threat detection instead of relying solely on reactive approaches. Third, machine learning-based systems can be trained to distinguish between normal network behaviour and malicious activities, reducing false positives and improving attack detection accuracy (Fathima et al., 2023). Selecting the right features for machine learning models to detect DDoS attacks is challenging. Feature engineering is crucial but has limitations. While machine learning algorithms perform well, they require lengthy training, are not real-time, and are computationally expensive, leading to potential misclassifications, and making them impractical for real-time implementation.

Traditional security mechanisms like encryption and authentication alone are insufficient, as they fail to detect sophisticated intrusion attempts. The absence of real-time, adaptive threat detection methods further exacerbates security risks. Motivated by these challenges, this research aims to develop a machine learning-based intrusion detection and classification system tailored for MANETs. Unlike conventional security solutions, the proposed system leverages supervised learning techniques, including Decision Trees, Random Forest, and an ensemble-based approach, to improve attack detection accuracy while minimizing computational overhead. This study provides a scalable and adaptive solution that enhances MANET security by integrating feature selection and optimizing model efficiency.

1.2 Motivation of the Study

One major problem for mobile ad-hoc networks is the need to ensure security. Due to its lack of centralised authority and scarce resources, MANET is more susceptible to security attacks. A secure transmission and communication mechanism must be established for MANET to function. This research is motivated by the increasing security threats faced by MANETs, the criticality of applications relying on MANETs, resource constraints of MANET devices, advancements in machine learning, the need for proactive threat detection, and the practical implementation of adequate security solutions. Moreover, it aims to develop adaptive security mechanisms to detect and mitigate real-time attacks and optimize resource utilization. By addressing these motivations, the research aims to enhance the security, resilience, and reliability of MANETs, ensuring their suitability for critical applications and contributing to the advancement of trustworthy MANET deployments.

1.3 Statement of Problem

Mobile Ad Hoc Networks (MANETs) are dynamic, decentralized networks with no fixed infrastructure, making them highly adaptable for various applications such as military operations, disaster response, and vehicular communication. However, their open and self-organizing nature makes them vulnerable to multiple security threats, including denial-of-service (DoS) attacks, black hole attacks, Sybil attacks, and wormhole attacks. Traditional security mechanisms, such as encryption and authentication, are inadequate in addressing these evolving threats due to the high mobility of nodes, the lack of centralized control, and the resource constraints of MANET devices. Existing Intrusion Detection Systems (IDS) struggle with high false alarm rates, limited adaptability to new attack patterns, and excessive computational overhead, creating a significant research gap in securing MANET environments. There is a critical need to enhance the security of MANETs by developing effective attack detection mechanisms. These mechanisms should detect and mitigate attacks in real time while considering the resource-constrained nature of MANETs (Joardar et al., 2023).

Traditional rule-based and signature-based approaches are often inadequate due to the dynamic nature of attacks and the evolving threat landscape. Hence, there is a demand for innovative and adaptive solutions that can effectively detect and respond to emerging attack patterns (Rahman, 2025). (KONJIT, 2006) research explored an IDS for MANETs but relied on signature-based detection, which is ineffective against unknown attacks and requires frequent updates. Similarly, other related studies have focused on static rule-based models or centralized IDS, which fail to adapt to the dynamic nature of MANETs. There is a lack of comprehensive research integrating adaptive machine learning techniques with efficient feature selection methods to improve detection accuracy and computational efficiency.

This study addresses these limitations by developing an adaptive, machine learning-based IDS tailored specifically for MANET environments. Unlike previous works, the proposed approach leverages an ensemble of classifiers, including Decision Tree, Random Forest, and Support Vector Machine (SVM), to enhance intrusion detection performance. The novelty of this research lies in its integration of feature selection techniques to optimize detection efficiency while minimizing computational overhead. Additionally, the study evaluates the model's scalability for deployment in real-world MANET scenarios. The absence of an automated, real-time intrusion detection mechanism leads to severe practical issues, including frequent network disruptions,

reduced communication reliability, and increased susceptibility to cyberattacks. By implementing an ML-based IDS, this study contributes to strengthening MANET security by providing a more adaptive, scalable, and computationally efficient solution.

1.4 Research questions

The research question includes:

1. How does an ensemble ML approach improve intrusion detection performance in MANETs?
2. How can feature selection improve detection accuracy?
3. What trade-offs exist between accuracy, false alarm rate, and computational efficiency in ML-based IDS for MANETs?

1.5 Objectives

1.5.1 General Objective of the Study

The general objective of this study is to develop and evaluate machine learning models for effective intrusion detection and classification in Mobile Ad Hoc Networks (MANETs).

1.5.2 Specific Objectives of the Study

To achieve this, the study pursued the following specific objectives:

- To identify and analyse specific security threats and attack patterns in MANETs.
- To select appropriate machine learning algorithms for attack detection in MANETs.
- To compare ML models based on accuracy and false alarms.
- To develop an adaptive ML-based IDS for MANETs.

1.6 Significance of the Study

This study is significant as it enhances the security of Mobile Ad Hoc Networks (MANETs), crucial in various applications like disaster recovery, military operations, and IoT systems. By applying machine learning techniques to intrusion detection and classification, the research offers a more efficient, adaptive, and scalable solution than traditional methods, which struggle with the dynamic nature of MANETs. The findings of this study could provide valuable insights for developing robust intrusion detection systems capable of detecting a wide range of attacks in real time and improving the overall security and reliability of MANETs. Additionally, this work contributes to the growing body of knowledge on machine learning applications in network security, offering a foundation for future research in this domain.

1.7 The Scope of the Study

The study uses machine learning techniques to detect and classify intrusions in Mobile Ad Hoc Networks (MANETs). It utilizes the UNSW-NB15 dataset, a benchmark dataset for intrusion detection, which was collected in 2015 and contains modern attack patterns relevant for network security research. From the dataset's 2.5 million records, 56,821 were used for training and 24,352 for testing to ensure computational feasibility. However, as the dataset does not capture real-time network traffic, some newer attack patterns may not be fully represented. A diverse range of attack types has been selected to maintain relevance to modern security threats. The study adhered to ethical research guidelines, ensuring no real-world malicious attacks were conducted. The dataset is publicly available and legally accessible for academic research, and privacy regulations such as the General Data Protection Regulation (GDPR) and Ethiopian cybersecurity guidelines have been followed, ensuring that no personally identifiable information (PII) was included. The proposed ML-based Intrusion Detection System (IDS) is primarily designed for simulated MANET environments rather than large-scale real-world deployments. The model is computationally lightweight and applicable in local industries such as military communication networks, disaster response teams, and Vehicular Ad Hoc Networks (VANETs). Although the model achieves high detection accuracy, real-time adaptability remains a challenge. Future research should explore incremental learning techniques to enable continuous learning from new network traffic patterns.

1.8 Limitations of the Study

Acknowledging the study's limitations enhances its credibility and provides avenues for future research improvements. The key limitations include dataset constraints, as the UNSW-NB15 dataset was initially collected in a wired network environment, while MANETs operate in a dynamic, wireless setting. Some MANET-specific attacks, such as wormhole and Sybil attacks, were not explicitly labeled in the dataset. A stratified random sampling technique was used to manage computational costs and memory constraints, selecting 56,821 records for training and 24,352 for testing. Due to the limited processing capabilities of MANET nodes, deep learning models such as CNNs and RNNs were not used despite their potential for improved accuracy. Instead, the study focused on lightweight classifiers such as Decision Trees, Random Forest, and Support Vector Machines (SVM) for efficiency.

The study was conducted in a simulated environment, meaning real-world deployment in an actual MANET network has not yet been tested. Challenges such as dynamic topology changes, low-bandwidth communication, and energy constraints were not fully addressed in the current model and require further optimization. The UNSW-NB15 dataset was chosen due to its diverse attack types compared to alternatives like NSL-KDD, making it more comprehensive for intrusion detection research. Public availability ensures reproducibility and comparability with existing studies. The dataset size was reduced to assure feasibility for real-time IDS deployment on resource-constrained MANET nodes while maintaining a balanced attack-to-normal traffic ratio.

Unlike traditional networks with centralized security infrastructure, MANETs require lightweight, distributed detection mechanisms due to their self-organizing nature. This study addresses the challenge by proposing an ensemble-based IDS model that enhances classification performance while minimizing computational overhead. To further improve the effectiveness and scalability of the proposed model, future research should explore real-time learning and adaptability using incremental or online machine learning, integration with real-world MANET testbeds to evaluate practical feasibility, incorporation of newer datasets to account for evolving cybersecurity threats, and optimization of resource usage to ensure compatibility with low-power MANET devices. This study provides an effective machine learning-based IDS framework for MANETs while acknowledging certain dataset limitations, computational constraints, and deployment challenges. The proposed model is scalable for simulation-based environments and applicable in local industries such as defense and emergency response. Future improvements should enhance real-time adaptability and testing in practical MANET settings.

1.9 Organization of the Paper

This thesis is structured into six chapters. Chapter 1 explains the background of the study, motivation, statement of the problem, objectives, and significance of the study. Chapter 2 reviews the literature on machine learning, MANETs, and datasets. Chapter 3 outlines the materials and methods for detecting intrusion attacks using machine-learning algorithms. Chapter 4 describes the proposed intrusion detection and classification methods in detail and offers a thorough implementation and assessment of the proposed techniques. Chapter 5 includes the experimentation and subsequent discussion. Finally, Chapter 6 summarises the research findings and suggests directions for future work.

CHAPTER TWO

LITERATURE REVIEW AND RELATED WORKS

2.1 Introduction

Artificial intelligence techniques have been actively employed for security solutions in recent years due to their robust automated feature extraction methods. Researchers have extensively utilised AI and machine learning in the past decade to detect network intrusions, proposing various learning-based techniques to enhance effectiveness. This section provides a comprehensive assessment of recent work, exploring the latest algorithms and approaches to identify areas for improvement in intelligent network intrusion detection systems regarding cost-effectiveness, efficiency, and accuracy.

2.2 AD Hoc Network

Laptops have recently become desktop computers because of continuous portability, convenience, and storage advancements. Users can now travel freely on battery power thanks to small computers' gigabyte-level storage, high-resolution displays, wireless capabilities, and pointing devices. Devices can interact without depending on pre-existing wired infrastructure in wireless ad hoc networks, such as MANETs. These networks allow devices to create independent groups that can choose to join fixed networks. The ability of ad hoc networks to organize themselves independently without requiring centralized management is a crucial feature. In these networks, terminals can act as routers, end-systems, and intermediate systems. Using intermediate nodes that function as routers, two nodes not directly in communication range can exchange data in Mobile Ad-hoc Networks. Because of this characteristic, multi-hop wireless networks are another name for ad hoc networks (Mohsin, 2022). Adaptable to changing traffic and conditions, ad hoc networks dynamically form among wireless nodes, offering flexibility compared to fixed cellular infrastructure. Nonlinear signal attenuation in wireless media allows ad hoc networks to achieve better energy efficiency than fixed cellular networks potentially. Energy-efficient routing and power resource management are optimized in ad hoc networks due to dynamic connections and adaptability. Increased spatial reuse in ad hoc networks enhances network capacity and spectral efficiency, overcoming interference challenges faced by fixed base stations in cellular networks. Ad hoc networks with adaptability and spatial reuse, contribute to superior capacity and spectral efficiency, especially in dynamic communication scenarios. Well-

suited for pervasive communication, ad hoc networks align with heterogeneous networks, offering dynamic device communication without relying on fixed infrastructure (Sharma & Rashid, 2020). The Mobile Ad hoc Network is vulnerable to denial-of-service attacks due to its complex routing methods, making its topology more volatile than other networks. MANET's unique characteristics include network flexibility and mobile node mobility. The author (Srivastava et al., 2021) discussed network speed, latency, and packet transfer rate, proposing using a neural network called a jump field for packet transmission to improve performance metrics. Machine learning applications are suggested to prevent packet loss and recover from network congestion.

A training dataset is necessary to train the models in a particular training mode and detect wormhole assaults using machine learning. This dataset can be created through experiments for classification purposes or gathered from real-time settings. The training dataset includes descriptive features that specify the properties of the data along with a target value. The machine learning algorithms can learn the patterns and relationships required for wormhole attack detection thanks to these attributes and target values. Machine learning techniques like K-Nearest Neighbor (KNN), Support Vector Machine (SVM), Decision Tree (DT), Linear Discrimination Analysis (LDA), Naive Bayes (NB), and Convolutional Neural Network (CNN) are used to counter wormhole attacks, which attack the network layer and interfere with routing protocols (Abdan & Seno, 2022).

The author (Meddeb et al., 2023) employed autoencoders and Deep Neural Networks to detect anomalies in Mobile Ad Hoc Networks. Compared to a Deep Autoencoder, it presents a Stacked Autoencoder for Dimensionality Reduction that exhibits higher accuracy in identifying malicious nodes. Real-time anomaly detection and flexibility in response to shifting network conditions are among its strong points. Nonetheless, the impact of an unbalanced dataset is recognized, and it is stressed that more study is required to test the method against increasingly complex threats. Although encouraging, resolving dataset problems and carrying out a more thorough comparative analysis would strengthen the validity of the suggested approach. Mobile ad hoc networks are efficient but vulnerable to attacks due to their open-access medium and dynamic topology. Existing intrusion detection systems (IDS) datasets are limited for evaluating mobile networks. To overcome this, an SVM classifier automatically generates rules for effective intrusion detection in mobile networks.

According to research by Apruzzese et al. (2022), Intrusion Detection is a new security approach for existing computers and data networks. Intrusion Detection Systems (IDS) focus on host audit trails and network traffic analysis to detect potential attacks in real time. Despite promising concepts, prevention-based network security has limitations, including the impossibility of achieving perfect security due to design flaws. Crypto-based systems are ineffective against specific attacks and lack sufficient protection against insider misuse.

The study (Poongothai & Jayarajan, 2018) suggested using a multi-level classification technique to improve mobile ad hoc network intrusion detection. It combines an enhanced multiclass SVM method with a tree classifier that uses labelled training data for binary classification. Data reduction is accomplished by choosing specific attributes from the KDD 99 cup dataset. The experimental results demonstrate improvements in false alarm and detection rates. The method divides assaults into four categories: root-to-location (R2L), user-to-root (U2R), denial of service (DOS), and probing. It does this by combining SVM and Decision Tree. The method seeks to improve accuracy, shorten training and testing periods, and protect MANETs from intrusion attacks.

2.3 Applications of MANET

Mobile Ad-hoc Networks (MANETs) are decentralized networks composed of mobile devices that communicate with each other without a fixed infrastructure. MANETs have various applications across domains due to their flexibility, adaptability, and dynamic nature. Some notable applications of MANETs include (Alo et al., 2018),

2.3.1 Military Communications

In military scenarios, Mobile Ad-hoc Networks are used to rapidly deploy communication networks without relying on fixed infrastructure. They offer dynamic deployment, redundancy, and adaptability to changing conditions. MANETs support tactical manoeuvring, provide stealth and security, and are scalable for diverse military operations. They enable real-time information sharing, reduce dependence on infrastructure, and enhance interoperability among various communication devices used by military personnel (Alshammari & Al-Shehri, 2019).

2.3.2 Emergency and Disaster Response

Mobile Ad-hoc Networks (MANETs) are crucial in disaster-stricken areas due to their ability to establish rapid, infrastructure-independent communication networks. They enable rescuers and emergency teams to coordinate efforts, share critical information in real-time, adapt to

challenging terrains, and provide reliable communication even without traditional infrastructure. MANETs enhance the efficiency of rescue operations in dynamic and unpredictable disaster scenarios (Olusanya & Vincent, 2020).

2.3.3 Search and Rescue Operations

In wilderness search and rescue operations, Mobile Ad-hoc Networks enable communication between search teams in challenging terrains without fixed infrastructure. MANETs facilitate real-time coordination, adapt to dynamic environments, and provide reliable information sharing, enhancing the efficiency of search and rescue efforts (S. S. Anjum et al., 2017).

2.3.4 Vehicular Ad-hoc Networks (VANETs)

MANETs are an integral part of VANETs used in intelligent transportation systems. Vehicles with communication devices can form a dynamic network to share information about traffic conditions and road hazards and enhance road safety. Vehicular ad Hoc Networks (VANETs), utilizing the principles of Mobile ad Hoc Networks (MANETs), are essential for intelligent transportation systems. VANETs contribute to collision avoidance and efficient traffic management and enable coordinated emergency responses, making transportation systems more intelligent and connected (Kanellopoulos & Cuomo, 2021).

2.3.5 Crowd Sensing and Monitoring

MANETs enable crowd-sensing applications where mobile devices form a dynamic network to collect and share real-time information about the environment. This is valuable for monitoring air quality, weather conditions, and other environmental factors, fostering community engagement, and providing scalable, rapid deployment solutions for diverse sensing needs (AbualSaud et al., 2019), (Al Sawafi et al., 2020).

2.3.6 Smart Agriculture

Mobile Ad Hoc Networks (MANETs) are crucial in precision agriculture, providing flexible and dynamic communication among sensors, drones, and agricultural equipment. This enables real-time monitoring, adaptability to changing field conditions, optimisation of farming practices, reduced infrastructure dependency, enhanced precision in resource application, and increased efficiency in responding to emerging issues. MANETs contribute to improved productivity and sustainability in agriculture (Tripathy et al., 2021), (Alghazzawi et al., 2021).

2.3.7 Sensor Networks in Remote Areas

Sensor networks in remote areas are crucial for diverse applications, including environmental monitoring, wildlife tracking, precision agriculture, disaster management, infrastructure monitoring, energy management, communication infrastructure, long-term research, and humanitarian aid. These networks, comprised of distributed sensors, collect and wirelessly communicate data. They contribute significantly to understanding ecosystems, conserving wildlife, optimizing agriculture, providing early disaster warnings, ensuring infrastructure safety, managing energy resources, establishing communication links, supporting research, and aiding humanitarian efforts. Overall, sensor networks are vital in addressing unique challenges in isolated regions, enhancing understanding, sustainability, and well-being. Mobile Ad Hoc Networks (MANETs) are well-suited for deploying sensor networks in remote or hostile environments due to their infrastructure independence, dynamic self-organizing nature, and rapid deployment capabilities. They enable environmental monitoring, wildlife tracking, scientific data collection, and precision agriculture, all while reducing infrastructure costs (Tripathy et al., 2021).

2.3.8 Education and Training

Mobile Ad Hoc Networks (MANETs) are valuable for collaborative learning in education, enabling dynamic collaboration, resource sharing, real-time discussions, flexible learning spaces, enhanced engagement, project collaboration, instant feedback, mobile learning opportunities, and a cost-effective solution for creating temporary networks in educational settings. MANETs enhance the overall educational experience for students and teachers (Safari et al., 2023).

2.3.9 Social Networking

MANETs are crucial for social networking in areas with limited infrastructure, allowing direct connections, on-the-go networks, proximity-based interactions, file sharing, real-time collaboration, event networking, offline social networking, community building, and emergency communication. They provide flexible and resilient solutions, enhancing connectivity in various scenarios (Tchembe & Tchoupe Tchendji, 2023).

2.3.10 IoT Communication

In IoT scenarios, MANETs offer a vital communication infrastructure for devices in dynamic environments and challenging deployments. They adapt to changes, ensuring seamless

connectivity among IoT devices, and addressing challenges where traditional networks may struggle to deploy (Alameri, 2018).

2.4 Protocol Layers of MANET

The OSI model, established by the International Organization for Standardization in the 1970s, consists of seven layers facilitating communication between diverse systems in large networks. This thesis focuses on the combined application, presentation, and session layers as the application layer. In contrast, the TCP/IP model, developed by DARPA in the late 1960s, dominates modern networks with its five layers (Ramphull et al., 2021).

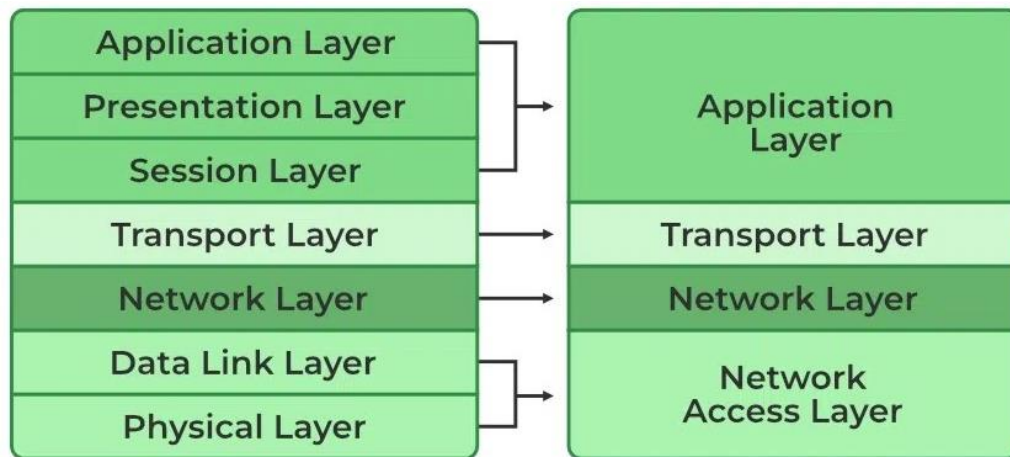


Figure 2.1: TCP/IP and OSI model

2.4.1 Physical Layer

Wireless communications involve electromagnetic waves, with technologies like radio frequency, infrared, and microwave in WLANs. In Mobile hoc networks (MANETs), nodes broadcast signals to others within their radio coverage, causing potential interference. Dense network topology improves connectivity but raises the risk of collisions, leading to packet loss and increased energy consumption. Every networking layer is enhanced, either by suggested modifications or standardization. Physical-layer Network Coding (PNC), a technique that was presented in (P. Chen et al., 2020), significantly increased multi-hop network capacity at the expense of a somewhat increased Bit Error Rate (BER).

In contrast to many existing systems, PNC proactively modifies transmission time slots to minimize collisions among neighbouring nodes while using wireless nodes' broadcast nature to increase throughput. The 802.11 standard families have been defined and enhanced throughout time by the vast technical professional association known as the Institute of Electrical and

Electronics Engineers (IEEE). The protocols comprising 802.11 standard families are listed in (IEEE Standards Association, 2010). Each 802.11 protocol's physical layer characteristics and actions are specified for the wireless networks' physical channel. The transmission mode, data rate, physical attributes, topology, synchronization, are all included in the specifications. A physical layer packet has the payload and all the overhead information needed for transmission and reception.

2.4.2 Data Link Layer

The data link layer comprises two sub-layers: Logical Link Control (LLC) and Media Access Control (MAC). The MAC sub-layer moves network layer data-grams between Network Interface Cards, while the LLC sub-layer handles error checking and flow control. The network layer datagram is encapsulated in a data link layer frame and transmitted through the physical layer. The 802.11 PHY standards use a uniform MAC layer with a consistent frame structure, including a header, body, and a 32-bit Frame Check Sequence (FCS) for error checking. The payload (body) can be up to 2312 bytes, and the 30-byte header contains essential information such as MAC addresses, sequence control, duration, power management, and encryption details. The 2-byte Frame Control field encapsulates control information outlined in Figure 2.3 (Bhatti & Saleem, 2020).

2	2	6	6	6	2	6	0 – 2312	4
Frame control	Duration ID	Destination address	Source address	Receiver address	Sequence control	Transmitter address	Payload	FCS

Figure 2.2: IEEE 802.11 frame format in bytes

2	2	4	1	1	1	1	1	1	1	1
Protocol version	Type	Subtype	To DS	From DS	More Fragments	Retry	Power management	More data	WEP	Order

Figure 2.3: IEEE 802.11 Frame control field in bits

The Distributed Coordination Function (DCF) with Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) protocol serves as the access mechanism for the 802.11 MAC layer. In order to handle both hidden and exposed terminal concerns, it is optional to use an expanded CSMA version with virtual carrier sensing via Request-to-Send (RTS) and Clear-to-Send (CTS) packets. Figure 2.5 shows the essential CSMA communications. Figure 2.4 illustrates the expanded version, which includes RTS/CTS control packets added expressly to address issues with exposed and hidden terminals. The RTS/CTS mechanism is absent from the basic version.

Collision avoidance is a joint effort between the MAC and physical layers. Using the 'listen before talk' principle, CSMA checks channel availability. Communication starts with the sender's RTS packet, conveying the destination and transmission duration. Upon receiving RTS, the receiver sends a CTS packet to neighboring nodes, specifying the sender's address and Network Allocation Vector (NAV) duration. Nodes store NAV locally, refrain from transmitting, and a receiving node may enter power-saving mode if NAV is nonzero.

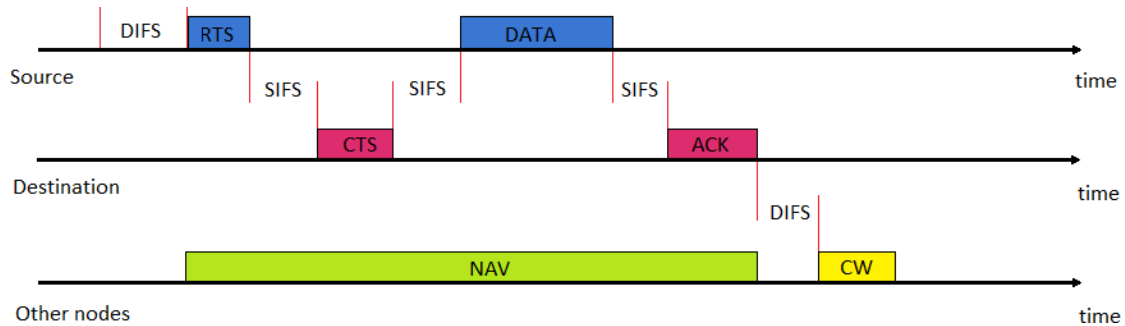


Figure 2.4: Transmission with RTS/ CTS

If the channel is busy, the node waits and selects a random back-off interval (contention window) before attempting to retransmit. The countdown occurs when the channel is idle, and the node transmits when the countdown reaches zero.

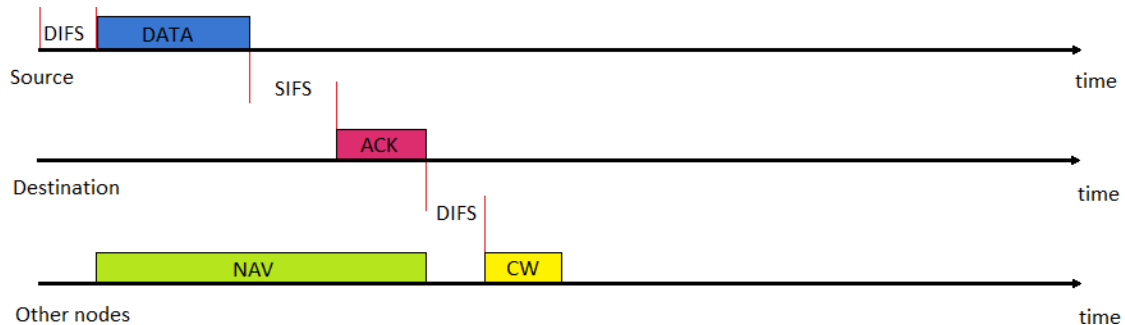


Figure 2.5: Transmission without RTS/ CTS

After sending a packet, the node waits for acknowledgement, which is critical in wireless networks with high bit error rates. Acknowledgements follow successful packet reception; if not received within a set time, the channel remains idle. The Exponential Random Backoff Mechanism adjusts contention dynamically, doubling after collisions. Acknowledgements introduce delays, but 802.11n's Block ACK enhances efficiency. Selective retransmission targets only lost packets in case of errors. In (Guo et al., 2022), the authors simulated a MANET and identified varying carrier sensing, transmission, and interference ranges. The authors suggested

an alternative mechanism to RTS/CTS to tackle the issues arising from this extended interference range.

2.4.3 Network Layer and Routing Protocol

The network layer, being the most researched and complex among layers, facilitates packet movement and communication between nodes. Routing, a critical function, builds paths in ad hoc networks for mobile nodes connected by wireless links. Nodes, behaving as hosts or routers, switch when forwarding packets in the correct direction. In mobile ad hoc networks, routing faces challenges like interference, asymmetrical links, and high packet loss due to mobility. Attacks on MANETs often target the network layer. Rapid topology changes and high mobility pose problems for routing algorithms, leading to delays, potential loops, and increased traffic. Detecting neighbouring nodes through HELLO messages can flood the network with broadcasts, causing congestion, packet loss, and energy depletion. Routing algorithms must address dynamic topology, convergence, unidirectional links, scalability, simplicity, power conservation, loop prevention, robustness, quality of service, low latency, packet order preservation, and support for priority, low overhead, and high throughput.

2.4.3.1 Routing Management

In mobile ad hoc networks, nodes move freely, leading to frequent changes in routing paths stored in local caches, known as routing tables. This mobility can quickly render routing tables outdated. To maintain up-to-date tables and ensure connectivity, nodes periodically broadcast route updates. This involves checking if neighbouring nodes are still present and updating the routing tables accordingly. Connectivity management in mobile ad hoc networks involves route discovery, selection, and maintenance.

Route discovery: In mobile ad hoc networks, nodes check routing tables for available paths before transmitting data. If no path exists, the route discovery mechanism is initiated, broadcasting a query packet with propagation restrictions to prevent flooding. Upon receiving the query, the destination sends a route reply back to the source along the discovered path, providing cost information. Optimization considerations include simplicity, overhead, and latency. Flooding is mitigated through mechanisms allowing intermediate nodes to reply if they know the path and an expanding ring search. Despite the importance of route discovery, challenges include managing network overhead. Various studies propose optimizations, which may result in unidirectional links or non-optimal routes.

Route selection: In a mobile ad hoc network, the routing algorithm selects the optimal path based on higher capacity, lower delays, lower error rates, lower interference, and more excellent route reliability. Unlike wired networks, which favour minimum hop count, wireless ad hoc networks prioritise shorter paths for increased reliability, given the higher likelihood of link failures and lower Signal-to-Noise Ratio (SNR) over longer distances.

Route maintenance: Due to topology changes, route maintenance is critical in wireless ad hoc networks. It involves detecting and addressing broken links, often caused by factors like battery depletion. If a node detects a broken link through unsuccessful retransmissions, it informs the source to choose an alternative path. The source either proceeds with retransmission using an alternative path from its routing table or initiates a route discovery if no alternative is available.

2.4.3.2 Topology-Based Routing

Topology-based routing employs three protocol categories: proactive, reactive, and hybrid, differing in the timing of path construction. Proactive routing relies on pre-existing path knowledge, requiring frequent updates through route discoveries. It offers immediate packet transmission but faces challenges like periodic network flooding and scalability issues in more extensive networks. Each node maintains a routing table, and in scenarios with frequent topology changes, convergence time may lead to packet drops (Agrawal et al., 2023).

In (Srilakshmi et al., 2022), the OLSR protocol is introduced for MANETs, serving as a proactive and link-state protocol with periodic control message exchanges throughout the network. OLSR optimises the traditional link-state protocol by reducing information size and flooding query packets. Unlike the classical link-state algorithm, OLSR employs a subset of neighbours known as multipoint relay selectors (MPRs) instead of every link, minimising flooding by retransmitting packets exclusively through MPRs. The protocol maintains scalability in large, dense wireless networks, avoiding additional overhead during link failures. Nodes identify neighbours by broadcasting HELLO messages to one-hop neighbours, where the HELLO message includes information about link status up to two hops away. Link status can be unidirectional, bidirectional, or MPR. Nodes strategically select one-hop neighbouring MPRs to cover every node within a two-hop range, aiming for optimality with a smaller MPR set. The MPR set is recalculated in response to detected topology changes. Nodes utilise Topology Control (TC) messages to announce their MPR selector set, periodically broadcasted to MPRs by the OLSR

algorithm. The TC message includes the neighbours of the transmitting MPR, enabling every node in the network to acquire routing knowledge.

DSDV (Destination-Sequenced Distance-Vector) is a dynamic routing protocol proposed for mobile ad hoc networks. It maintains routing tables at each node, using sequence numbers to prevent loops. Periodic updates include destination addresses, hop counts, and sequence numbers, optimizing paths. Broken links trigger updates with infinite metrics. Two packet types, full dump and incremental dump, manage update information and reduce network overhead in scenarios of low node mobility (Abbas et al., 2020). WRP (Wireless Routing Protocol) is a path-finding algorithm that solves the counting-to-infinity problem. It efficiently conveys the length and second-to-last hop of the shortest path to each destination. Nodes maintain tables, including distance, routing, link cost, and message retransmission. The algorithm constructs a shortest path spanning tree through neighbour interactions. Periodic HELLO messages are broadcasted for route maintenance, and in case of link failure, nodes broadcast update messages to establish new paths (Tang et al., 2020).

Dynamic Source Routing (DSR) is a reactive protocol for multi-hop wireless ad hoc networks, utilizing source routing to create loop-free routes. It employs Route Request messages for discovery and maintains routes in a cache. DSR allows asymmetric routes and supports unidirectional links, enhancing overall performance. Route maintenance involves acknowledgements and handling broken links. Additional features include preventing Route Reply storms, decrementing hop limits, salvaging packets, automatic route shortening, and spreading Route Error messages for improved performance and reliability (Khudayer & Anbar, 2020).

AODV (Basil et al., 2024) is a reactive protocol with a sequence number concept from DSDV. It creates routes on demand, using a broadcast route discovery mechanism similar to DSR. AODV dynamically establishes route table entries at intermediate nodes, specifying the next hop for each destination. It relies on sequence numbers and broadcast IDs to prevent loops and uses unsolicited Route Replies for broken links. Route maintenance involves locally broadcast HELLO messages, updating tables, and detecting broken links. Unlike DSR, AODV lacks support for asymmetric links and introduces additional overhead with frequent HELLO messages.

A flat architecture is suitable for smaller networks, but the hierarchical architecture becomes more advantageous as the number of nodes increases, reducing overall overhead. This approach, detailed in (Krishnamoorthy et al., 2023), involves a hierarchical cluster-based Intrusion Detection System (IDS) that prioritizes mobility and energy awareness. Detection responsibilities are distributed to nodes with higher battery capacity, emphasizing the importance of preserving battery life for node longevity and network security.

The cluster-based routing protocol (Krishnamoorthy et al., 2023) classifies nodes as cluster heads, cluster members, or undecided, managing bidirectional and unidirectional links. Nodes exchange HELLO messages containing neighbour and cluster adjacency tables. Cluster formation designates the lowest ID node as the cluster head. Route discovery involves flooding, but Route Request packets are forwarded only to cluster heads. Routes include cluster heads and gateways. Route maintenance and optimization mechanisms are similar to flat architectures, handling Route Error packets, salvaging routes, and optimizing routes using 2-hop information. CBRP operates distributed, emphasizing cluster formation, adjacent cluster discovery, and routing processes.

2.4.3.3 Position-Based Routing

Distance routing effect algorithm for mobility optimizes wireless communication by considering the relative motion between nodes. It efficiently uses bandwidth and energy, ensuring robustness against link failures and maintaining multiple routes with an 80% packet delivery rate. The protocol employs directed flooding, minimizing control message sizes, and adapting retransmission frequency and hop count based on distance and mobility for enhanced accuracy (Hmeed et al., 2023).

2.4.3.4 Energy Aware Routing

Localized energy-aware routing (Abdullah, 2021) is an extension of the DSR protocol where mobile nodes decide to forward Route Request messages based on their remaining battery level. If the battery level exceeds a threshold, the message is forwarded, impacting the chances of reaching the destination or finding an energy-efficient path. If no reply is received, the source initiates a new request with an increased sequence number, prompting intermediate nodes to lower the threshold for message relaying.

The Alternate Path Routing protocol is employed for load balancing and survivability in ad hoc environments. It balances energy consumption by utilizing various routes, known as alternate

route sets, from source to destination. The quality of alternate route sets is crucial, emphasizing the need for decoupled routes to avoid single points of failure. APR can use proactive routing, with the source node aware of alternate routes incurring additional overhead, or reactive routing, determining alternate routes upon request (Bhardwaj & El-ocla, 2020).

2.4.4 Transport Layer

The transport layer, positioned fourth in the TCP/IP suite, offers end-to-end services, including congestion control. TCP is the dominant protocol in the Internet (handling around 90% of data), while UDP is simpler and used for multimedia. Originally designed for wired networks, TCP faces challenges in wireless ones due to dynamic topology changes. TCP ensures reliable, connection-oriented data delivery but misinterprets issues as congestion, impacting performance in ad hoc networks. Proposed enhancements provide partial solutions. Despite limitations, TCP is used in Mobile Ad Hoc Networks (MANETs) for compatibility with various applications and integration with wired infrastructures (Mast et al., 2023).

TCP, designed for wired networks, encounters issues in wireless environments. It misinterprets link failures as congestion, leading to bandwidth reduction and extended retransmission timeouts. Errors from interference and fading trigger mechanisms that further decrease transmission rates. Channel contention and delays caused by contention, route changes, and errors impact performance. TCP's reliance on Round-Trip Time (RTT) faces challenges in MANETs with sudden delay variations, contributing to assumed packet losses and subsequent retransmissions, reducing the transmission rate. The protocol lacks effective mechanisms to address these issues, significantly impacting performance. Along with its enhancements, TCP falls short of delivering a comprehensive transport layer. While enhancements address specific issues, their combination may introduce compatibility challenges.

Ad Hoc Transport Protocol (ATP) is a specialised and reliable transport protocol designed for ad hoc networks. It employs layer coordination, rate-based transmissions, and decoupled congestion control and reliability mechanisms. ATP collaborates with lower layers to estimate transmission rates, manage congestion, and detect route breaks. The protocol uses rate-based transmissions with feedback from intermediate nodes and periodic selective ACKs for acknowledgement. ATP is incompatible with TCP, limiting its use with TCP-dependent applications, and requires cooperation from intermediate nodes and lower layers. The default transmission frequency of selective ACKs in ATP is one second, potentially causing delayed detection of packet losses

(Naguib et al., 2018). ACTP is a lightweight transport protocol placing reliability control in the application layer. It offers a middle-ground performance between TCP and UDP in MANETs, allowing dynamic priority assignment for different applications. Incompatible with TCP, ACTP lacks a congestion control mechanism, making it scalable but prone to congestion issues (Parween et al., 2023).

2.4.5 Application Layer

The application layer facilitates communication between end systems, supporting diverse applications such as the web, file sharing, Voice over IP, and more. Security measures, including cryptographic techniques and Intrusion Detection Systems (IDS), are crucial at this layer (Kurumbanshi, 2023).

2.5 Security Attacks in MANET

MANETs are self-configuring networks of mobile devices that communicate with each other without the need for a centralised infrastructure. However, their decentralised nature makes them vulnerable to attacks, resulting in slow network performance, message delays, and uncontrolled traffic. By being aware of these attacks, network administrators and security professionals can implement appropriate countermeasures to protect MANETs and ensure the secure transmission of data. Attacks are commonly categorised into four types based on their objectives and methods. These four types are (Ali et al., 2023):

- **Passive Attacks:** Passive attacks gather information without altering or disrupting the network or its data. The attacker aims to eavesdrop on communication and gain unauthorised access to sensitive information. Examples of passive attacks include sniffing network traffic, capturing passwords, or intercepting confidential data.
- **Active Attacks:** Active attacks involve the alteration or disruption of network communication. The attacker actively modifies or destroys data, injects malicious code, or disrupts the normal functioning of the network. Examples of active attacks include denial of service (DoS) attacks, man-in-the-middle attacks, and packet spoofing.
- **Insider Attacks:** Insider attacks occur when a trusted entity within the network, such as an authorised user or an employee, abuses its privileges to compromise the network's security. Insider attacks can involve unauthorised access, data theft, or intentional disruption of network operations.

- **Distribution Attacks:** Distribution attacks exploit network infrastructure or software vulnerabilities to compromise multiple nodes or the entire network. These attacks can propagate rapidly and affect a large number of devices. Examples of distribution attacks include worms, viruses, botnets, and malware (Marteau, 2019).

In this context, it is essential to explore the basic classes of attacks that can impact the performance and reliability of MANETs. Common attack types that can occur within a MANET environment (Gupta et al., 2022):

- **Denial of Service (DoS) Attacks:** These attacks aim to disrupt the normal functioning of a network by overwhelming it with a flood of illegitimate requests or exploiting vulnerabilities in network protocols. In a MANET, DoS attacks can consume network resources, making the network slow or unusable.
- **Black Hole Attacks:** In a black hole attack, a malicious node falsely advertises itself as having the shortest path to a destination node. As a result, it attracts data traffic, but instead of forwarding it, the malicious node drops the packets, causing message delays and data loss.
- **Grayhole Attacks:** Like black hole attacks, grayhole attacks involve a malicious node selectively dropping or modifying packets. The difference is that, in grayhole attacks, the malicious node may only drop a portion of the packets, making it difficult to detect the attack.
- **Wormhole Attacks:** In a wormhole attack, two or more colluding malicious nodes create a tunnel between them, allowing them to forward packets over a much shorter distance than expected. This attack can lead to significant delays and disrupt the usual routing process.
- **Sybil Attacks:** Sybil attacks involve the creation of multiple fake identities by a single malicious node. The attacker can manipulate routing protocols and disrupt the network's operation by pretending to be multiple nodes.
- **Jamming Attacks:** Jamming attacks involve deliberate interference with wireless communication signals, causing disruption or blocking of legitimate network traffic. These attacks can severely impact the performance and reliability of a MANET.

Recent research on the UNSW-NB15 dataset improves intrusion detection systems (IDS) using advanced machine learning techniques and an optimized deep CNN-LSTM model, improving

classification accuracy (Thaljaoui, 2025). The work by (Sultan et al., 2023) proposed a deep learning ANN model for detecting DoS attacks, while another used stacked autoencoders to improve feature representation in IDSs. Table 2.1 summarizes the types in the UNSW-NB15 dataset.

Table 2.1: Summary Table of Attack Types in UNSW-NB15

Attack Type	Primary Goal	Common Methods	Impact
DoS	Disrupt services	Traffic flooding, protocol exploitation	System downtime, resource exhaustion
Exploits	Take advantage of vulnerabilities	Buffer overflows, privilege escalation	Unauthorized access, system takeover
Fuzzers	Discover security flaws	Malformed inputs, automated testing	System crashes, vulnerability discovery
Generic	Bypass security mechanisms	Brute-force, cryptographic attacks	Unauthorized access, credential leaks
Reconnaissance	Gather information	Port scanning, traffic analysis	Preparation for future attacks
Worms	Self-propagation	Exploit vulnerabilities, automated replication	Widespread infection, network disruption
Analysis	Extract sensitive data	Packet sniffing, log analysis	Data leaks, intelligence gathering
Backdoor	Maintain unauthorized access	Malware, rogue admin access	Persistent control, data theft

2.6 An intrusion detection system

An Intrusion Detection System (IDS) is a critical component of network security infrastructure designed to detect and respond to potential security breaches and malicious activities. The development of IDS technology arose from the need to protect computer networks from increasing cyber security threats. The concept of intrusion detection dates back to the 1980s when computer networks became more prevalent (Kemmerer & Vigna, 2002). Early intrusion detection systems focused on monitoring system logs and detecting specific patterns or signatures associated with known attacks. These systems were rule-based, relying on predefined patterns to identify malicious activities.

As computer networks evolved and became more complex, traditional security measures like firewalls and antivirus software became insufficient to address the growing threat landscape. Intruders found innovative ways to exploit vulnerabilities and bypass traditional security controls. This led to the development of more advanced IDS technologies. Modern IDS solutions incorporate a combination of signature-based detection, anomaly detection, and behavior-based analysis techniques. Signature-based detection involves comparing network traffic or system events against a database of known attack signatures. Anomaly detection focuses on identifying deviations from established norms and models of normal behaviour. Behaviour-based analysis looks for suspicious activities or patterns that may indicate an attack. IDS can be deployed in various network environments, including enterprise networks, data centers, and cloud-based infrastructures. They can operate in different modes, such as network-based IDS (NIDS), host-based IDS (HIDS), or a combination of both (hybrid IDS). NIDS monitors network traffic in real-time, while HIDS monitors activities on individual hosts or servers. Over time, IDS technology has evolved to include more advanced features such as machine learning, artificial intelligence, and threat intelligence integration. These advancements enable IDS to better detect and respond to sophisticated and evolving cyber threats (Lata & Singh, 2022).

2.6.1 Intrusion Detection and Classification System in MANET

Intrusion Detection and Classification System (IDCS) in Mobile Ad Hoc Networks (MANETs) is an important security mechanism to identify and classify malicious activities in a MANET environment. MANETs are self-configuring networks consisting of mobile nodes without any centralised infrastructure. Due to the dynamic nature of MANETs and the absence of a centralised authority, securing these networks is challenging. Two main categories of (IDS) are applied to MANET (Abbood et al., 2023).

1. Distributed and Cooperative Intrusion Detection System

A Distributed and Cooperative Intrusion Detection System (DCIDS) is a security mechanism that involves the collaboration of multiple nodes in a network to detect and respond to intrusions. Unlike a centralised intrusion detection system, where a single node or central authority performs all the detection and analysis tasks, a DCIDS cooperatively distributes the detection and analysis tasks among multiple nodes. In a DCIDS, each node in the network acts as an intrusion detection sensor and collects local information about network traffic, node behaviour, and other relevant data. The nodes analyse the collected information using local intrusion detection algorithms or

techniques to identify potential intrusions. Instead of making detection decisions independently, the nodes collectively collaborate and share their findings to make more accurate intrusion detection decisions. The cooperation in a DCIDS can take various forms, including:

- **Information Sharing:** Nodes in the DCIDS share their local detection results, such as detected anomalies or suspicious activities, with other nodes in the network. This information-sharing allows nodes to collectively analyse and correlate the data to make more informed intrusion detection decisions.
- **Collaborative Decision-Making:** Nodes in the DCIDS use the shared information to make decisions collectively about the presence of intrusions. This can involve voting mechanisms or consensus algorithms to determine whether a detected event is a genuine intrusion or a false positive.
- **Distributed Data Fusion:** Nodes in the DCIDS fuse the locally collected data and detection results to create a global view of the network's security status. This data fusion helps identify attacks that span multiple nodes or exhibit global patterns.

The benefits of a DCIDS include improved detection accuracy, enhanced resilience to node failures, and the ability to detect attacks that may be difficult to identify using a single sensor. By leveraging multiple nodes' collective knowledge and capabilities, a DCIDS can detect and respond to a wide range of intrusions in a distributed network environment. Designing and deploying a DCIDS involves addressing several challenges, such as ensuring secure communication and information exchange among nodes, managing the overhead of information sharing and collaboration, and handling the scalability of the system as the network size grows.

2. Hierarchical Intrusion Detection System

A Hierarchical Intrusion Detection System (HIDS) is an intrusion detection system architecture that organises the detection and analysis tasks into a hierarchical structure. In a HIDS, multiple detection components are arranged hierarchically, allowing for efficient and scalable intrusion detection in large and complex networks. The key idea behind a HIDS is to divide the detection tasks into different levels or tiers based on the network's structure and characteristics. Each level focuses on specific aspects of intrusion detection, and the results are aggregated and analysed at higher levels to make more informed decisions. The hierarchical structure enables efficient distribution of detection workload, better resource management, and improved accuracy. The typical components or levels in a HIDS are:

- **Sensor Level (Lower Level):** At the lowest level of the hierarchy, individual sensors or agents are deployed throughout the network. These sensors monitor network traffic, node behaviour, and other relevant data to detect potential intrusions. They use local intrusion detection techniques, such as signature-based, anomaly, or behaviour-based, to analyse the collected data and identify suspicious activities or patterns.
- **Intermediate Level:** The intermediate level in the hierarchy receives and aggregates the detection results from multiple sensors. It performs data fusion and correlation to identify potential attacks that span multiple sensors or exhibit global patterns. By combining the information from different sensors, this level focuses on reducing false positives and improving the overall detection accuracy.
- **Manager Level (Higher Level):** The manager receives the aggregated detection results from the intermediate level and performs further analysis and decision-making. It applies advanced analysis techniques, such as machine learning algorithms or rule-based systems, to classify and prioritise the detected intrusions. The manager can generate alerts, trigger response actions, or provide recommendations to network administrators or security personnel.

The hierarchical structure of a HIDS offers several advantages:

Scalability: The distribution of detection tasks across multiple levels allows for efficient scaling in large networks. Each level can handle a subset of the network, reducing the overall detection workload and improving system performance.

Efficient Resource Utilization: By distributing the detection tasks, the computational and communication resources can be utilised more efficiently. Local detection at the sensor level reduces the data transmitted to higher levels, minimising the network overhead.

Improved Detection Accuracy: Aggregating and correlating higher-level detection results helps reduce false positives and improve intrusion detection accuracy. The hierarchical structure allows for a more comprehensive analysis of network-wide activities.

Flexibility and Modularity: A HIDS's hierarchical architecture allows for flexibility in adding or removing detection components at different levels without disrupting the overall system. This enables the integration of new detection techniques or the adaptation to evolving threats.

2.6.2 Intrusion Detection and Classification Models in the MANET

Intrusion Detection Models in Mobile Ad Hoc Networks (MANETs) are designed to detect and mitigate intrusions and malicious activities within the dynamic and decentralized nature of MANETs. Several intrusion detection models have been proposed for MANETs, considering the unique characteristics of these networks. Here are a commonly used models (Sworna et al., 2023):

- **Cooperative-based Intrusion Detection Model:** In this model, MANET nodes collaborate to collectively detect and report intrusions. Each node monitors its behaviour and the behaviour of its neighbouring nodes. Nodes exchange information and cooperate to identify deviations from normal behaviour. Cooperative-based models leverage techniques like voting, reputation systems, or distributed decision-making to reach a consensus on intrusion detection.
- **Hierarchical Intrusion Detection Model:** A hierarchical model organizes nodes in the MANET into hierarchical clusters or levels. Each level has intrusion detection capabilities, such as monitoring, analysis, and decision-making. The detection results are aggregated and analyzed at higher levels to make informed decisions. Hierarchical models provide scalability, resource management, and improved detection accuracy by distributing the detection tasks.
- **Trust-based Intrusion Detection Model:** Trust-based models focus on establishing trust relationships among nodes in the MANET. Nodes assess the trustworthiness of their neighbours based on past interactions, behaviour, or recommendations from other trusted nodes. Trust values determine the likelihood of a node being involved in malicious activities. Trust-based models can help identify and isolate malicious nodes from the network.
- **Game-theoretic Intrusion Detection Model:** Game theory models consider MANET security a strategic game between nodes. Each node makes decisions based on its objectives and the actions of other nodes. Game-theoretic models analyze the interactions and strategies of nodes to identify potential attacks or malicious behavior. These models explore Nash equilibrium, cooperation, and incentive mechanisms to detect and mitigate intrusions.

- **Machine Learning-based Intrusion Detection Model:** Machine learning models utilize data-driven techniques to detect intrusions in MANETs. These models are trained on a dataset of normal and malicious network behavior and learn to classify network activities based on patterns and features. Machine learning algorithms, such as decision trees, support vector machines, or neural networks, can be used to build intrusion detection models that adapt to evolving threats.
- **Signature-based Intrusion Detection Model:** Signature-based models use a database of known attack signatures or patterns to detect intrusions. These models compare network traffic or events against the signature database and raise an alert if a match is found. Signature-based models effectively detect known attacks but may have limitations in detecting unknown or novel attacks.

These intrusion detection models can be combined or adapted based on the specific requirements and characteristics of the MANET. The choice of the model depends on factors such as scalability, resource constraints, network dynamics, and the types of attacks expected in the MANET environment (Tran et al., 2022).

2.7 Machine Learning Model

Machine learning algorithms can improve network security by automating calculations and decisions, such as packet classification and anomaly detection (Eltanbouly et al., 2020). Machine learning algorithms are typically categorized into the following types:

2.7.1 Supervised Learning

In supervised learning, the algorithm learns from labeled training data, where each instance is associated with a known class or label. The algorithm generalizes from the training data to predict or classify new, unseen data. In network security, supervised learning algorithms can be trained to classify network traffic, detect specific types of attacks, or identify patterns associated with malicious activities (Danilchenko et al., 2023).

2.7.2 Unsupervised Learning

Unsupervised learning algorithms work with unlabeled data, meaning no predefined class labels exist. These algorithms aim to find patterns, structures, or anomalies in the data without prior knowledge of the expected outcomes. In network security, unsupervised learning algorithms can be used for anomaly detection, identifying unusual network behaviour, or clustering similar network traffic patterns (Kushwah & Garg, 2024).

2.7.3 Semi-Supervised Learning

Semi-supervised learning combines elements of supervised and unsupervised learning. It uses a small amount of labeled data and a more significant amount of unlabelled data to train the algorithm. The labeled data provides some guidance or supervision to the algorithm, while the unlabeled data helps it discover additional patterns or structures. Semi-supervised learning can be useful in network security when labeled data is limited or costly to obtain (Yao et al., 2019).

2.7.4 Reinforcement Learning

Reinforcement learning involves training an algorithm to make sequential decisions in an environment to maximize a reward signal. The algorithm learns through trial and error, receiving feedback in the form of rewards or penalties based on its actions. Reinforcement learning algorithms can be applied in network security for tasks such as adaptive intrusion response, dynamic firewall rule optimization, or network resource allocation (Mayadunna et al., 2017).

2.7.5 Deep Learning

Deep learning is a subfield of machine learning that utilizes artificial neural networks with multiple layers to learn hierarchical representations of data. Deep learning algorithms like deep neural networks can automatically learn complex features and patterns from raw input data. In network security, deep learning algorithms can be used for tasks like intrusion detection, malware detection, or traffic classification (Sultan et al., 2023).

2.8 Dataset

NSL-KDD Dataset

The NSL-KDD dataset stands out in intrusion detection and network security, serving as an enhanced iteration of the original KDD Cup 99 dataset. Its refinement aims to overcome the limitations and challenges present in the earlier version. Designed primarily to assess the performance of intrusion detection systems (IDS), the dataset encompasses network traffic data that features both regular and diverse intrusive activities.

Comprising a blend of categorical and numerical features extracted from network traffic, the dataset delineates various facets of network connections, encompassing protocol type, service, source and destination IP addresses, among others. Binary labels indicate whether a network connection is normal or anomalous (intrusive). The dataset encompasses multiple attack types, including DoS (Denial of Service), R2L (Unauthorized access from a remote machine), U2R (Unauthorized access to a local superuser), and probing. The NSL-KDD dataset is commonly

partitioned into training and testing sets for the convenience of training and testing intrusion detection systems. Addressing limitations present in its predecessor, the KDD Cup 99 dataset, such as redundant records and a focus on specific attack types, the NSL-KDD dataset is modified to offer a more authentic portrayal of contemporary network traffic (Su et al., 2020).

UNSW-NB15 Dataset

Similar to NSL-KDD, UNSW-NB15 has been developed to assess the effectiveness of intrusion detection systems (IDS). Its goal is to offer a contemporary and lifelike depiction of network traffic and potential attacks. The dataset comprises diverse features extracted from network traffic, encompassing source and destination IP addresses, port numbers, connection durations, and various protocol-related details. Binary labels are assigned to the UNSW-NB15 dataset, indicating whether a network connection is standard or malicious. The dataset encompasses a broad spectrum of attack types, including Denial of Service (DoS), reconnaissance, and probing. The dataset is typically divided into training and testing sets to facilitate developing and evaluating intrusion detection models (Zoghi & Serpen, 2023).

CSE-CIC-IDS2018 Dataset

The goal of the CSE-CIC-IDS2018 dataset was to create a diverse instructional dataset designed explicitly for intrusion detection on network systems.

It was developed in 2018 in collaboration with the Communication Security Establishment (CSE) and the Canadian Institute of Cyber Security (CIC). The authentic user behaviours seen on networks are mirrored in this dataset. It includes seven types of attacks: online attacks, botnet operations, heartbleed exploits, brute-force attempts, infiltration techniques, Distributed Denial of Service (DDoS), and denial of service (DoS). Only a portion of the dataset was used because of its size. For evaluation reasons, CSV files of Friday network traffic, specifically on 02-03-2018 and 16-02-2018, were selected. Segments that most closely resembled actual attack scenarios were chosen due to the dataset's notable class imbalance. The selected subset concentrates on four main classes: botnet activity, low-level HTTP DoS attacks, DoS-Hulk attacks, and regular network traffic. These classes were combined to create a single 'attack' class. Upon removing null entries and duplicate records, 1,074,342 records were left in the dataset, of which 290,089 were deemed harmful (Leevy & Khoshgoftaar, 2020). Table 2.1 highlights several issues in related works, categorized according to classification type, dataset selection, computational resource requirements, training complexity, and inadequate results on minority

classes. These problems are discussed to illuminate the shortcomings observed in the reviewed literature.

2.9 Related Works

Recently, machine learning algorithms have achieved significant success in predicting DDoS attacks. A feature selection approach described by (Yusof et al., 2017) enhances intrusion detection systems through machine learning by combining DDoS Characteristic Features (DCF) with Consistency Subset Evaluation (CSE). Researchers (Igbe et al., 2017) introduced the Dendritic Cell Algorithm (DCA), an AIS-based method for detecting frequent denial of service and distributed DoS attacks that disrupt network communication. According to (Mahmodi et al., 2018), a method using a genetic algorithm (GA) was proposed to identify DDoS attacks on cloud platforms, optimizing the Bernoulli Naïve Bayes (BNB) classifier. The Semi-Supervised approach (Idhammad et al., 2018) involves entropy estimation, co-clustering, and information gain ratio (IGR) for feature selection, with the Extra-Trees ensemble algorithm for classification. Network traffic data entropy is analyzed over time-based sliding windows; followed by co-clustering into three groups when entropy limits are reached, feature selection using IGR, and classification with Extra-Trees ensemble. Authors (M. Anjum & Shreedhara, 2019) proposed a method to enhance performance over traditional supervised and unsupervised techniques for DDoS detection by introducing a Semi-Supervised Machine Learning technique that merges both approaches. According to the author (Mukhametzyanov et al., 2019) highlighted that neural networks (NN) are effective for DDoS detection, using the Deduct or modelling environment to develop a single-layer perceptron NN model with 35 neurons (11 input, 23 hidden, and 1 output). The accuracy of this model was evaluated using a contingency table. According to (Verma et al., 2020) a hybrid method for feature selection and classification, which uses Mean Absolute Deviation (MAD) thresholding combined with random forest (RF) classification, was found to be highly effective. The researcher (Das et al., 2019) proposed a network IDS (NIDS) that detects DDoS attacks using ensemble classifiers with a reduced feature dataset. In (Ma et al., 2020) major challenges in developing IoT intrusion detection systems were addressed, and a novel CNN model was introduced, utilizing feature fusion and a cross-entropy loss function with multilayer convolution, surpassing current deep learning methods focused on typical network intrusions. (Prathyusha & Kannayaram, 2021) described using artificial immune systems (AIS) for detecting and mitigating DDoS attacks in cloud computing. Authors (Bhardwaj et al., 2020) recommended an architecture for DDoS classification combining autoencoders (AE) and deep neural networks (DNN). Initially, a

simple AI and DNN model was developed with randomly set hyperparameters, followed by an enhanced version incorporating additional algorithmic improvements. (KONJIT, 2006) introduced an IDS for MANETs based on signature-based detection techniques. While effective for known attack patterns, the study's approach was limited in detecting emerging or novel threats. Additionally, the study did not explore machine learning-based solutions, leaving a gap in adaptive security mechanisms. The author (Bagyalakshmi, 2020) proposed two approaches: a filter method using Learning Vector Quantization (LVQ) and a dimensionality reduction method using Principal Component Analysis (PCA). Naïve Bayes (NB), Decision Tree (DT), and Support Vector Machine (SVM) were employed to classify DDoS attacks, utilizing the features selected by each method. Table 2.2 illustrates a summary of the related work.

Table 2.2: Summary of the Related Work

aper author	Title	Method/Algorithm	Results	Paper Limitation
(Dora & Lakshmi, 2022)	Optimal feature selection with CNN feature learning for DDoS attack detection using metaheuristic-based LSTM	CNN and LSTM	Accuracy 96.37%, Precision 97.44%, Recall 97.43%, F1 score 97.87	Specially made for detecting DDoS attacks
(Perez-Diaz et al., 2020)	A Flexible SDN-Based Architecture for Identifying and Mitigating Low-Rate DDoS Attacks Using Machine Learning	The CIC DoS 2017 dataset was used with RT, REP Tree, Random Forest, and SVM algorithms.	Accuracy Random Forest 94.41% and SVM 93.10%	Collecting traffic features in short intervals increases overhead.
(Xue Chen, Giulia Tessari, Massimo Fabris, 2021)	Using Machine Learning for DoS Attacks Diagnostics	Support Vector Machines (SVM), Random Forest (RF), and K-Nearest Neighbors (KNN)	Accuracy 92% and precision 93%	To handle large-scale network data and improve the interpretability of models.

(Ma et al., 2020)	A Deep Learning-Based DDoS Detection Framework for the Internet of Things	CNN model with feature fusion and cross-entropy loss	Accuracy 92.99%	The CNN model for IoT intrusion detection faces challenges related to training complexity and the need for extensive computational resources.
(Yusof et al., 2017)	Adaptive Feature Selection for Denial of Services (DoS) Attack	Feature selection DCF + CSE	Accuracy 91.7%	The DDoS Characteristic Features (DCF) and Consistency Subset Evaluation (CSE) approaches may fail to handle varying attack patterns or adapt to new attack types.
(Mahmodi et al., 2018)	Using Genetic Algorithm to Improve Bernoulli Naïve Bayes Algorithm to Detect DDoS Attacks in Cloud Computing Platform	Genetic Algorithm (GA) and Bernoulli Naïve Bayes (BNB) classifier	Accuracy 82.44%	The GA-based optimization of BNB may face convergence and complexity issues, limiting its real-time detection practicality.
(Idhammad et al., 2018)	Semi-supervised machine learning approach for DDoS detection	Semi-supervised approach and Extra-Trees ensemble algorithm	Accuracy 98.23%	The Semi-Supervised approach with entropy, co-

				clustering, and extra trees may struggle with feature selection accuracy and adapting to evolving attacks.
(M. Anjum & Shreedhara, 2019)	Performance Analysis of Semi-Supervised Machine Learning Approach for DDoS Detection	Semi-Supervised Machine Learning Combination of supervised and unsupervised techniques	Accuracy 93.26%	The semi-supervised technique may not fully use labelled data and could be affected by training data quality and quantity.
(Mukhametzyanov et al., 2019)	The Neural Network Model of DDoS Attacks Identification for Information Management	Neural network modelling and Single-layer perceptron (NN)	Accuracy 97.94%	The single-layer perceptron model is limited by its simplicity, affecting its ability to capture complex attack patterns.
(Verma et al., 2020)	An Adaptive Threshold-Based Attribute Selection to Classify Requests Under DDoS Attack in Cloud-Based Systems	Hybrid feature selection and classification Mean Absolute Deviation (MAD) thresholding and Random Forest (RF) classification	Accuracy 98.23%	MAD thresholding and random forest hybrid method struggle with specific thresholds and manual tuning, impacting generalization.

(Prathyusha & Kannayaram, 2021)	A cognitive mechanism for mitigating DDoS attacks using the artificial immune system in a cloud environment	Artificial Immune Systems (AIS)	Accuracy 96.7%	AIS for DDoS detection in cloud computing might struggle with dynamic and sophisticated attack patterns.
(Bhardwaj et al., 2020)	Hyperband Tuned Deep Neural Network with Well Posed Stacked Sparse AutoEncoder for Detection of DDoS Attacks in Cloud	Autoencoder (AE) and Deep Neural Network (DNN)	Accuracy 98.43%	AE and DNN may face challenges with hyperparameter tuning and require extensive computational resources.
(Jemili et al., 2023)	Intrusion Detection based on Ensemble Learning for Big Data Classification	Naive Bayes (NB), XGBoost, Decision Tree (DT), and Random Forest (RF) were evaluated for intrusion analysis and data classification.	The combination of Random Forest and XGBoost achieved the highest accuracy of 97%, while Random Forest alone had an accuracy of 96.58%.	Lack of specific details of big data in intrusion detection or how ensemble learning addresses the limitations of simple learning methods.
(Prasad et al., 2020)	Intrusion Detection in Ad Hoc Network	Numerous machine-learning techniques have been employed	Multilayer Perceptron in supervised	The training process was time-consuming, and the

	Using Machine Learning Technique	for the classification of benign and malicious packet information.	learning achieves 98.5% detection and 4.7% false alarm rates for blackhole attacks.	dataset used in the study was simulated using NS2 rather than real-time data.
--	----------------------------------	--	---	---

2.10 Summary

The study fills a gap in the related works by presenting a new end-to-end machine-learning strategy for enhanced network intrusion detection. This method uses the most recent intrusion datasets for assessment and is lightweight and economical. The proposed approach distinguishes itself from other recent research by being straightforward, efficient in processing, capable of managing multi-class categorisation, and not requiring complicated computing resources.

Collecting traffic features in short intervals increases overhead, and handling large-scale network data remains challenging. The CNN model for IoT intrusion detection faces high computational demands, while DCF and CSE approaches may struggle with varying attack patterns. GA-based BNB optimization has convergence issues, and the semi-supervised approach may suffer from feature selection inaccuracies and reliance on training data quality. The single-layer perceptron model is too simplistic for complex attacks, and the MAD-random forest hybrid requires manual tuning. AIS struggles with sophisticated cloud-based attacks, while AE and DNN face hyper-parameter tuning challenges.

Furthermore, the proposed approach shows significant benefits over methods based on machine learning. The study contributed significantly to the network intrusion detection of MANET by providing a simple, effective, and economical method that achieves results comparable to those of deep learning techniques, possibly resolving some drawbacks of deep learning-based methods.

CHAPTER THREE

MATERIALS AND METHODOLOGY

3.1 Introduction

This section presents a machine-learning approach for anomaly detection in Mobile Ad hoc Networks (MANETs). The initial steps involved preparing a dataset and implementing chosen machine learning algorithms.

3.2 Research Design

This study adopts a machine learning-based approach for intrusion detection and classification in Mobile Ad Hoc Networks (MANETs). The research follows a structured methodology that includes dataset acquisition, feature selection and extraction, model training, evaluation, and performance analysis. The proposed system is designed to detect and classify network intrusions using supervised learning techniques while ensuring computational efficiency and adaptability to dynamic MANET environments. This study adopts a machine learning-based approach for intrusion detection and classification in Mobile Ad Hoc Networks (MANETs). The research follows a structured methodology, including dataset acquisition, feature selection and extraction, model training and evaluation, and performance analysis. The proposed system is designed to detect and classify network intrusions using supervised learning techniques, focusing on computational efficiency and adaptability to dynamic MANET environments.

3.3 Research Framework

This study presented a structure for classifying and forecasting intrusion utilizing machine learning techniques on an established dataset. A structured research process was followed to achieve the research objective outlined in this thesis, as illustrated in Figure 3.1. The study comprises three main phases, detailed below:

Phase One: The research begins with an extensive literature review aimed at identifying and comprehensively understanding the problem. Various sources are examined to gain insights into the specific issue under investigation.

Phase Two: This phase focuses on data preparation. A total of 81,173 datasets are labelled and assigned to their respective categories. Data augmentation techniques are also applied using Python code to enhance model training.

Phase Three: In this phase, preprocessing and machine learning techniques are implemented to predict customer complaint varieties. The model's performance is evaluated by comparing accuracies, and it is tested using hidden DoS attacks from the datasets.

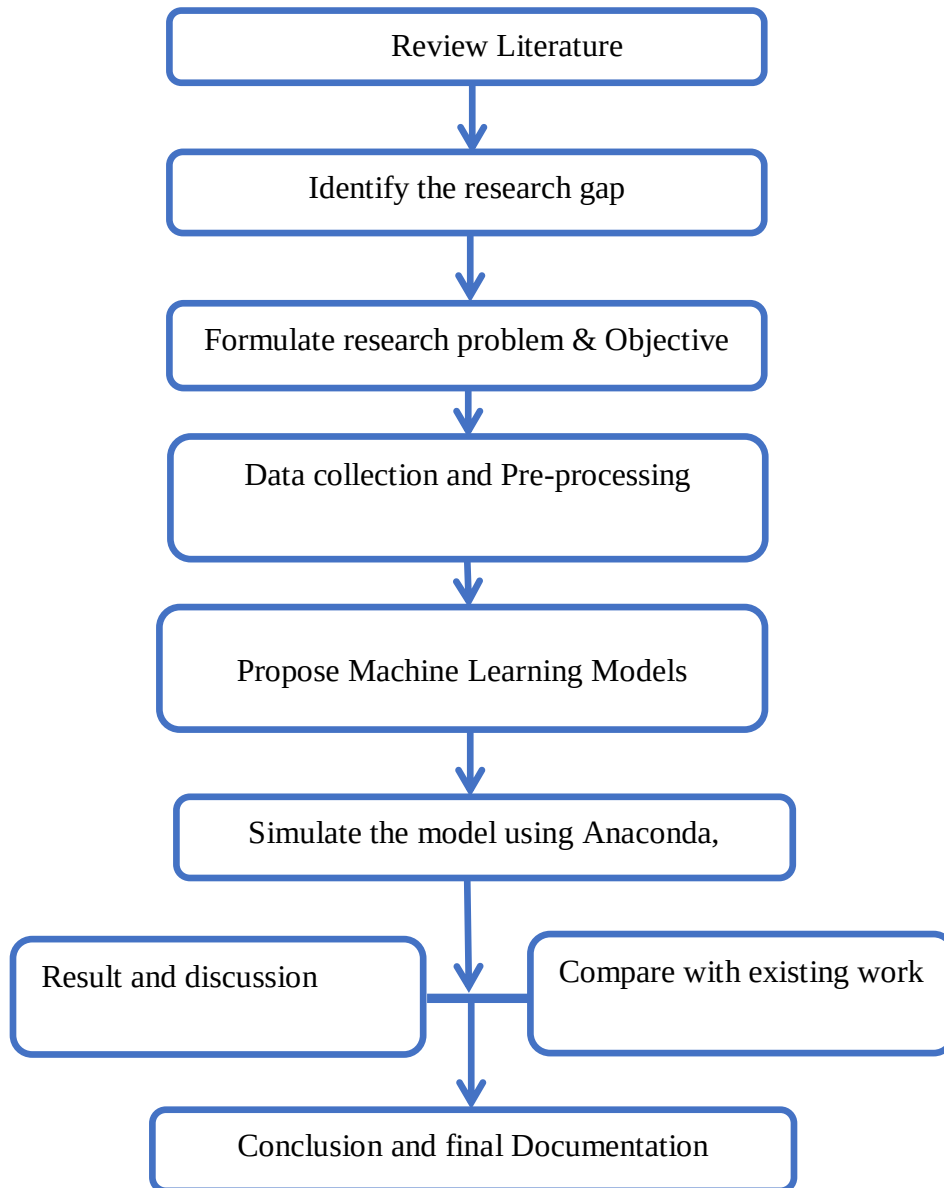


Figure 3.1: Research Framework

3.4 Data Pre-processing

Data pre-processing is preparing and cleaning raw data before analysis or modelling. It involves handling missing values, removing duplicates and outliers, integrating data from multiple sources, transforming data into a suitable format, selecting relevant features, discrediting data if

needed, addressing class imbalance, and scaling numerical features. These techniques help improve data quality, consistency, and compatibility with machine learning algorithms.

3.4.1 Feature Normalization

Data normalization is necessary for building a strong machine-learning classifier. It involves standardizing the data features to remove biases. By scaling the values to a range of 0 to 1, all numerical values are transformed to a standard scale. This improves the classifier's performance and allows fair comparison between different features. There are various normalization techniques, but two common ones are.

Min-Max Scaling (Normalization): This method scales the data to a fixed range, usually between 0 and 1. It subtracts the minimum value of the feature and then divides it by the range (the maximum value minus the minimum value). The formula for Min-Max Scaling is:

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (3.1)$$

Z-score Normalization (Standardization): This method standardizes the data by transforming it to have a mean of 0 and a standard deviation of 1. It subtracts the mean of the feature and then divides by the standard deviation. The formula for Z-score Normalization is:

$$X_{std} = \frac{X - \mu}{\sigma} \quad (3.2)$$

Where, X is the original feature value, X_{norm} or X_{std} is the normalized or standardized values respectively, X_{max} and X_{min} are the minimum and maximum values of the feature respectively, μ is the mean of feature and σ is standard deviation of the feature.

The normalization process does not negatively impact the classifier's performance by features with high numerical values. Feature normalization helps ensure that all features contribute equally to the analysis, preventing certain features from dominating based solely on their scale.

3.4.2 Feature Selection

Feature selection is choosing the most relevant features from a dataset to improve model performance and reduce complexity. Techniques include filter methods (like correlation analysis), wrapper methods (like forward selection), embedded methods (like Lasso regression), and dimensionality reduction techniques (like PCA). In light of this, only essential features for classification were retained. Recursive Feature Elimination (RFE), a backward feature selection method, was applied. RFE involves iteratively removing less important features by retraining models on various feature sets until optimal performance is achieved, determining the subset

used for prediction. The final optimal subset trains the ultimate algorithm. RFE conducts a greedy search for the best feature subset. However, not all models pair with RFE due to feature-sample imbalances. RFE with RF is selected for two reasons: RF maintains variables in the prediction equation and its ensemble advantage. RF also evaluates the importance of features internally (Q. Chen et al., 2018).

3.4.3 Hyperparameters Values

Hyperparameter values were employed to tackle over-fitting, and basic and uncomplicated architectures were trained and tested for machine learning models, as they are prone to over-fitting on training data.

3.5 Proposed Machine Learning Models for Intrusion Detection

In this research study, five separate machine learning models were trained. The datasets were obtained from public sources, normalized and then reduced using RF-RFE. The reduced datasets were split into training and testing sets. Traditional machine learning algorithms were initially employed, but their performance was limited due to the complex and high-dimensional data. To address this, a hybrid ensemble algorithm was proposed and trained on pre-processed datasets to classify attack types. Additional details are provided in the following sections.

3.5.1 Decision Tree Model

Mobile Ad Hoc Networks (MANETs) are particularly vulnerable to intrusion attacks, posing significant challenges in ensuring secure communication. Machine learning, especially decision tree algorithms, provides a robust solution for detecting and classifying these attacks. The decision tree algorithm, a supervised learning method used for classification and regression tasks, operates by dividing data into subsets based on the values of input features, forming a tree-like structure of decisions. This approach is highly interpretable and can process numerical and categorical data, making it well-suited for identifying patterns indicative of intrusion attacks. In this algorithm, internal nodes represent features of the dataset, branches denote decision rules, and each leaf node signifies the classification outcome. Detailed breakdown of the mathematical model used in decision tree algorithms,

Decision Tree Structure

A decision tree can be represented as a hierarchical structure consisting of:

- Note: These represent the features/attributes.
- Edges: These represent the outcome of a decision rule applied to the feature.

- Leaves: These represent the final decision or classification.

Splitting Criterion

The decision tree algorithm splits the dataset at each node based on a criterion to maximize the purity of the subsets. Common criteria include,

Gini Impurity:

$$Gini(D) = 1 - \sum_{i=1}^C p_i^2 \quad (3.3)$$

Where p_i^2 is the proportion of instances of class (i) in dataset (D), and (C) is the total number of classes.

Information Gain:

$$IG(D, A) = Entropy(D) - \sum_{v \in Values(A)} \frac{|D_v|}{|D|} Entropy(D_v) \quad (3.4)$$

Where (IG (D, A)) is the information gain of attribute (A), (D) is the dataset, (D_v) is the subset of (D) for which attribute (A) has value (v), and (Values(A)) are the possible values of attribute (A).

Splitting the Dataset

The dataset (D) is split based on the attribute (A) that maximizes the chosen splitting criterion.

For a given node, the best attribute (A*) is chosen as:

$$A^* = \arg \max_A IG(D, A) \quad (3.5)$$

Recursion

The algorithm recursively applies the splitting process to each subset (D_v). The recursion continues until one of the stopping criteria is met:

- All instances in a subset belong to the same class.
- No more attributes are available.
- A predefined depth limit is reached.
- A subset size falls below a certain threshold.

Predicting the Target Variable

The final decision tree is used to classify new instances. The decision rules dictate that the tree be traversed from the root to a leaf. The class label at the leaf node predicts the instance's outcome.

3.5.2 Random Forest Model

Random Forest is an ensemble learning model that builds multiple decision trees during training and outputs the mode of the classes (classification) or the average prediction (regression) of the individual trees. It leverages the concept of bagging (Bootstrap Aggregating) and random feature selection to improve predictive accuracy and mitigate overfitting. The Random Forest algorithm involves several components, including decision trees, bagging, and random feature selection.

Decision Trees: A decision tree is a flowchart-like structure in which each internal node represents a test on a feature, each branch represents the test's outcome, and each leaf node represents a class label (in classification) or a continuous value (in regression).

Bagging (Bootstrap Aggregating): Bagging is a technique used in Random Forests to create multiple subsets of the training data by sampling with replacement. Each decision tree in the Random Forest is trained on a different subset of the data, allowing for diversity in the ensemble.

Random Feature Selection: Random Forests consider only a random subset of features for splitting at each decision tree node. This random feature selection ensures that each tree in the forest is trained on different subsets of features, reducing the correlation between trees and improving the diversity of the ensemble.

- Bootstrap Sampling: Random Forests create multiple subsets of the training data using bootstrap sampling (sampling with replacement).
- Random Feature Selection: At each decision tree node, only a random subset of features is considered for splitting. This helps decorrelate the trees and improve the diversity of the ensemble.
- Ensemble Learning: Random Forests aggregate the predictions of multiple decision trees through voting (for classification) or averaging (for regression).

The model for a Random Forest can be summarized as follows:

Training Phase:

- Randomly select subsets of the training data with replacement (bagging).
- For each subset, train a decision tree using a random subset of features at each node.
- Repeat the above steps to create multiple decision trees in the forest.

Training Procedure:

- Create (N) bootstrap samples from the training data.

- For each sample, train a decision tree, considering only a random subset of features at each node.
- Aggregate the predictions of all trees to make the final prediction.

Prediction Phase:

- For a new instance, let each decision tree in the forest independently predict the class label.
- Aggregate the predictions using a majority voting mechanism to determine the final prediction.

3.5.3 Support Vector Machine Model

Support Vector Machine (SVM) is a supervised learning algorithm that can be used for classification or regression tasks. SVM works by finding the hyperplane that best separates the data points of different classes in the feature space. The data points closest to the hyperplane are called support vectors, and they play a crucial role in defining the position and orientation of the hyperplane. SVM is capable of capturing complex patterns in data by using kernel functions to transform the input features into higher-dimensional spaces where linear separation is possible. A hyperplane in an n -dimensional space is an $n-1$ -dimensional subspace that divides the space into two halves. In a 2D space, it's a line; in a 3D space, it's a plane. Margin is the distance between the hyperplane and the nearest data points from either class, known as support vectors. SVM aims to maximize this margin. Support vectors are the data points that lie closest to the hyperplane and are the most difficult to classify. They are critical in defining the position and orientation of the hyperplane.

For a binary classification problem, given a training dataset $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$, the SVM aims to find a hyperplane $w \cdot x + b = 0$ that maximizes the margin.

If the data is linearly separable, the hard margin SVM given.

Optimization problem:

$$\min_{w,b} \frac{1}{2} ||w||^2 \quad (3.10)$$

$$\text{Subject to: } y_i(w \cdot x_i + b) \geq 1, \quad \forall i$$

For non-linearly separable data, the soft margin SVM allows for some misclassification by introducing slack variable ξ_i . Optimization problem,

$$\min_{w,b,\xi} \frac{1}{2} ||w||^2 + C \sum_{i=1}^n \xi_i \quad (3.11)$$

$$\text{Subject to: } y_i(w \cdot x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0 \quad \forall i$$

Here, C is regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error. For non-linearly separable data SVM uses the kernel trick to transform the data into a higher-dimensional space where a linear hyperplane can be used to separate the data. Common kernels include.

Linear kernel: $K(x_i, x_j) = x_i \cdot x_j$

Polynomial kernel: $K(x_i, x_j) = (x_i \cdot x_j + 1)^d$

Radial basis function (RBF) kernel: $K(x_i, x_j) = \exp(-\gamma ||x_i - x_j||^2)$

Sigmoid kernel: $K(x_i, x_j) = \tanh(\alpha x_i \cdot x_j + r)$

The optimization problem is often solved in its dual form, which is computationally more efficient for high-dimensional data.

Dual formulation:

$$\max_{\alpha} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j K(x_i, x_j) \quad (3.12)$$

Subject to:

$$\sum_{i=1}^n \alpha_i y_i = 0, 0 \leq \alpha_i \leq C, \forall_i$$

Here, α_i are the Lagrange multipliers.

Once the model is trained, the prediction for the new data point x is,

Decision function:

$$f(x) = \sum_{i=1}^n \alpha_i y_i K(x_i, x) + b \quad (3.13)$$

Class label:

$$\hat{y} = \text{sign}(f(x)) \quad (3.14)$$

3.5.4 K-nearest Neighbours Classification (KNN)

A simple algorithm classifies values based on similarity measures such as distance. It is used for both classification and regression predictive problems. However, it is primarily used in classification problems. This is simple and popular because it is easy to interpret the output. Similarly, calculation time is faster. KNN does not assume any underlying distribution for the data. It makes predictions based on the entire training dataset. It does not learn a model from the training data. Instead, it memorizes the training dataset and makes predictions during inference. The algorithm uses a distance metric to find the 'k' nearest neighbours. The typical distance metrics include,

Euclidean distance:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (3.15)$$

Manhattan distance,

$$d(x, y) = \sum_{i=1}^n |x_i - y_i| \quad (3.16)$$

Minkowski distance (a generalization of Euclidean and Manhattan distances),

$$d(x, y) = (\sum_{i=1}^n |x_i - y_i|^p)^{1/p} \quad (3.17)$$

Hamming distance (for categorical data),

$$d(x, y) = \sum_{i=1}^n 1(x_i \neq y_i) \quad (3.18)$$

Algorithm

Classification

1. *Choose **k**: Decide the number of neighbours (k).*
2. *Compute Distances: Calculate the distance between the new data point and all the points in the training set.*
3. *Identify Neighbours: Identify the **k** nearest neighbors based on the computed distances.*
4. *Vote: The new data point is assigned the most common class among its k nearest neighbors (majority voting) for classification.*

Regression

1. *Choose k: Decide the number of neighbors (k).*
2. *Compute Distances: Calculate the distance between the new data point and all the points in the training set.*
3. *Identify Neighbours: Identify the k nearest neighbors based on the computed distances.*
4. *Average: For regression, the prediction is the average of the target values of the k nearest neighbors.*

3.5.5 Ensemble Method

Ensemble learning leverages the strengths and weaknesses of various classifiers by integrating them. Each classifier may have proficiency in certain attack types while facing challenges with others. Through the combination of multiple weaker classifiers and the use of a voting algorithm, a robust classification model is developed. This strategy helps address individual algorithms' limitations, resulting in a more robust overall model. The proposed ensemble model integrates RF, KNN, and SVM classification models. The models were trained collectively using a hybrid approach that includes RF Recursive Feature Elimination (RF-RFE) for feature reduction and a

stacking ensemble method to combine individual predictions through a majority vote. This approach effectively reduces variance and enhances predictive accuracy.

3.6 Performance Evaluation Metrics

Performance metrics like accuracy, precision, recall, and F1 score measure effectiveness. Various evaluation techniques, from traditional ones like accuracy to advanced ones like precision and recall, are utilized.

Accuracy: Accuracy assesses the proportion of correctly identified cases, encompassing true positives and true negatives, relative to the entire population. It is computed by dividing the number of correctly predicted positive and negative samples by the total sample size. Accuracy measures the ratio of accurately predicted classes to the total samples evaluated.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \times 100 \quad (3.3)$$

Precision: Precision represents the ratio of correctly predicted positive samples to all samples that were predicted as positive. Mathematically, it is defined as:

$$\text{Precision} = \frac{TP}{TP+FP} \times 100 \quad (3.4)$$

Recall: Recall, also known as sensitivity or true positive rate, quantifies the proportion of all correctly identified positive samples to all positive ones. It measures the model's ability to correctly identify positive patterns among all positive instances.

$$\text{Recall} = \frac{TP}{TP+FN} \times 100 \quad (3.5)$$

F1-Score: The F1-score represents the harmonic mean of precision and recall. It combines precision and recall equally, calculating the harmonic average of their rates to provide a balanced assessment of the model's performance.

$$F1 - \text{Score} = \frac{2TP}{2TP+FP+FN} \times 100 \quad \text{Or} \quad F1 - \text{Score} = 2 \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \times 100 \quad (3.6)$$

CHAPTER FOUR

PROPOSED MODEL AND IMPLEMENTATION

4.1. Proposed Model Architecture and Design

The datasets used in this investigation came from open sources. After the input data was normalised, RF-RFE, a hybrid technique, was used to reduce the features. After that, the datasets were divided into training and testing sets and stored in their reduced form. At first, sophisticated machine learning techniques were used to comprehend attack classifications and behaviours. However, individual algorithms did not perform well enough because of the high dimensionality and complexity of the data. The suggested hybrid ensemble technique was trained on the pre-processed datasets to identify the different attacks to overcome this difficulty. The proposed approach is shown in Figure 4.1.

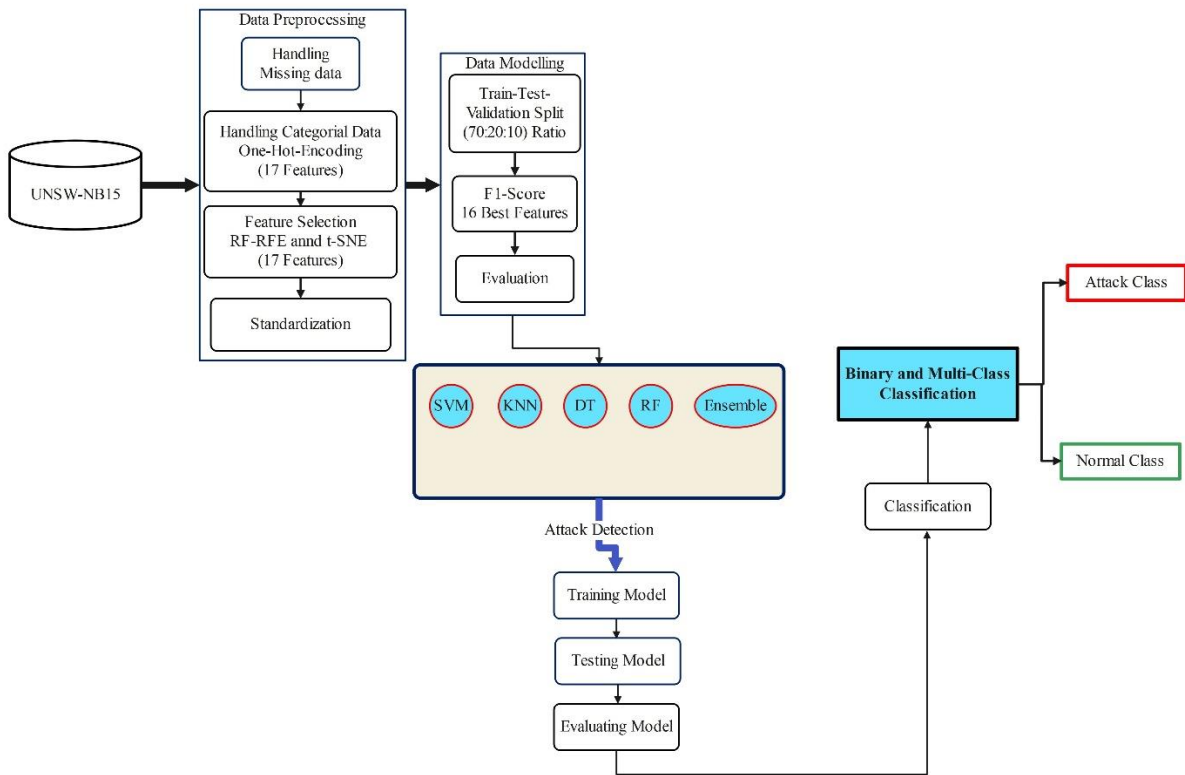


Figure 4.1: Proposed Intrusion Detection and Classification Approach

4.2 Description of Block Diagram

Pre-processing

- **Normalization:** This step ensures that all features are scaled within a specific range, which helps algorithms, particularly distance-based methods like KNN and SVM, to

perform better. For example, it might transform numerical values to a standard scale, preventing features with large ranges from dominating those with smaller ranges.

- **RF-RFE (Random Forest-Recursive Feature Elimination)** is a feature selection technique in which a Random Forest model ranks the importance of features. Then, Recursive Feature Elimination iteratively removes the least important features to improve model performance. This step reduces noise and computational load by keeping only the most relevant features.
- **t-SNE (t-distributed Stochastic Neighbour Embedding):** While RF-RFE is primarily used for feature selection, t-SNE visualizes high-dimensional data by reducing it to two or three dimensions. This can help visualize clusters of normal vs. intrusion data, which provides insights into the data distribution.

Train/Test Split

- **Training Data:** This subset of the data is used to train the machine learning models. The models learn patterns from this data to differentiate between regular and intrusion attack traffic.
- **Testing Data:** This subset is set aside during training to evaluate how well the trained model generalizes to unseen data. It is essential for validating the model's performance in a real-world scenario.

ML Algorithms

The diagram lists several machine learning algorithms that are applied to the pre-processed training data:

- **SVM (Support Vector Machine):** This algorithm finds the hyperplane that best separates data points of different classes. It's adequate for binary classification tasks like detecting normal vs intrusion traffic.
- **KNN (K-Nearest Neighbours):** This is a simple, instance-based learning method where classification is based on the majority label among the nearest neighbours of a data point.
- **DT (Decision Tree):** A tree-structured model that splits the data based on feature values to make decisions, leading to the final prediction.
- **RF (Random Forest):** An ensemble of decision trees, which improves prediction accuracy by reducing overfitting. It averages the results of multiple trees for robust predictions.

- **Ensemble:** This could involve combining the outputs of different models (e.g., through voting or averaging) to leverage the strengths of each algorithm and achieve better overall performance.

Trained ML Algorithms

After training, the machine learning models are ready to make predictions. These trained models are essential for real-time detection and mitigation of network intrusion attacks.

Evaluation

The effectiveness of the trained models is assessed using several performance metrics:

- **Precision:** This metric calculates how many of the optimistic predictions (intrusion detections) were correct, helping to minimise false positives.
- **Recall:** This measures the ability of the model to correctly identify all actual positive cases (intrusion attacks), aiming to minimise false negatives.
- **Accuracy:** The overall correctness of the model is measured as the ratio of correct predictions (both true positives and negatives) to the total number of predictions.
- **F1-Score:** A balanced metric that considers both precision and recall, giving a performance measure especially useful when dealing with imbalanced datasets.

4.3 Implementation of the Proposed Model

This section aimed to evaluate the effectiveness of our proposed system design, a key element of the broader framework, to understand its performance, strengths, and weaknesses. It presents a thorough experimental assessment of the proposed machine learning models for detecting and classifying intrusion in Mobile Ad Hoc Networks (MANETs). It details the implementation, dataset preparation, and training/testing methods.

4.3.1 Dataset Preparation

The study employs stratified random sampling to ensure diverse attack representation. The dataset comprises 2.5 million records, but a subset of 56,821 training records and 24,352 testing records was selected due to computational constraints. This subset maintains proportional attack class distribution while optimizing model training efficiency. The methodology ensures that the chosen data effectively represents normal and attack traffic patterns, improving generalization capability and enhancing detection performance. The study employs stratified random sampling to ensure diverse attack representation. The dataset comprises 2.5 million records, but a subset of 56,821 training records and 24,352 testing records was selected due to computational

constraints. This subset maintains proportional attack class distribution while optimizing model training efficiency. The methodology ensures that the chosen data effectively represents normal and attack traffic patterns, improving generalization capability Table 4.1 Provides a summary of UNSW-NB15 dataset statistics.

Table 4.1: UNSW-NB15 attack distribution

Attack category	Event count	Attack category	Event count	Attack category	Event count
Fuzzers	20,246	Analysis	2,677	Exploits	14,525
Recon	13,987	Backdoors	2,329	Generic	13,371
Normal	1,511	DoS	12,353	Worms	174

The UNSW-NB15 dataset contains various intrusion attack types. DoS disrupts services by overwhelming network resources while exploiting system vulnerabilities for unauthorized access. Fuzzers send malformed inputs to identify security flaws, and generic attacks use brute-force methods to bypass defenses. Reconnaissance gathers intelligence through scanning and analysis, often preceding other attacks. Worms self-replicate and spread across networks, whereas analysis attacks extract sensitive data. Lastly, backdoors provide hidden access, allowing attackers to bypass authentication. These attacks highlight the need for robust intrusion detection systems.

Table 4.2: Dataset description of UNSW-NB15 dataset

Rows	Columns
81,173	17

4.3.2 Dataset Feature Extraction

The UNSW-NB15 dataset contains 17 features, categorized into essential, content, time, and host features. Basic features include connection details like duration, protocol type, and bytes sent. Content features capture behaviours such as failed logins and root access attempts. Time features focus on traffic patterns, error rates, and connection statistics over time. Host features track destination host traffic, including the number of connections, service usage, and port usage. These features provide a comprehensive view of network traffic, aiding intrusion detection and classification tasks. These features can be used as target labels to train models to differentiate between attack types or as input features providing context about network traffic. For instance, distinguishing between DoS attacks and regular traffic or other network exploits requires a clear

understanding of these categorical distinctions. Combining these with numerical and binary features enhances the model's ability to perform detailed classification. Additionally, feature engineering can improve the dataset by creating new features that capture interactions between existing ones, such as the source-to-destination window size ratio, which may reveal network congestion indicative of DoS attacks. Temporal features that monitor changes over time, like variations in TCP, are valuable for detecting slow-lorries or other time-based attacks. Dimensionality reduction techniques, such as Principal Component Analysis (PCA), can focus the model on the most informative features, minimizing noise and enhancing generalization.

4.3.3 Experimental Setup

The experimental setup consists of multiple stages to evaluate the proposed intrusion detection framework comprehensively. The study utilizes the UNSW-NB15 dataset, which contains diverse attack types relevant to modern cyber threats. The dataset includes 2.5 million network traffic records, from which 56,821 records were selected for training and 24,352 for testing using stratified sampling to ensure balanced representation of attack types. Feature selection and extraction were conducted using Recursive Feature Elimination (RFE) with Random Forest (RF-RFE) to identify the most significant features, reducing computational overhead while maintaining high classification performance. The final feature set consists of 16 optimized features crucial for intrusion detection. The intrusion detection framework consists of a data preprocessing module, a feature selection module, a classifier module, and a decision-making module. The classifier module includes Decision Trees (DT), Random Forest (RF), Support Vector Machine (SVM), and an ensemble-based approach to improve detection accuracy. Hyperparameter tuning was performed using Grid Search Cross-Validation to optimize classifier performance, with key parameters selected based on empirical testing and domain expertise. The model's performance was evaluated using accuracy, precision, recall, F1-score, and confusion matrix analysis to ensure a comprehensive assessment of classification effectiveness. The experimental setup consists of multiple stages, providing an extensive evaluation of the proposed intrusion detection framework:

- **Dataset Acquisition:** The study utilizes the UNSW-NB15 dataset, which contains diverse attack types relevant to modern cyber threats. The dataset includes 2.5 million network traffic records, from which 56,821 records were selected for training and 24,352 for testing using stratified sampling to ensure balanced representation of attack types.

- **Feature Selection and Extraction:** The study employs Recursive Feature Elimination (RFE) with Random Forest (RF-RFE) to select the most significant features, reducing computational overhead while maintaining high classification performance. The final feature set consists of 16 optimised features crucial for intrusion detection.
- **Proposed Architecture:** The intrusion detection framework consists of a data preprocessing module, a feature selection module, a classifier module, and a decision-making module. The classifier module includes Decision Trees (DT), Random Forest (RF), Support Vector Machine (SVM), and an ensemble-based approach to improve detection accuracy.
- **Parameter Selection and Optimization:** Hyper-parameter tuning was performed using Grid Search Cross-Validation to optimise classifier performance. The key parameters for each classifier were selected based on empirical testing and domain expertise.
- **Performance Evaluation Metrics:** Accuracy, precision, recall, F1-score, and confusion matrix analysis were used to measure model performance, ensuring a comprehensive assessment of classification effectiveness.

The models were implemented using the Python programming language on a system with the following specifications: an 11th Gen Intel(R) Core (TM) i7-1165G7 @ 2.80GHz processor, 16GB RAM, and 1TB hard disk space. A 12GB NVIDIA Tesla K80 Graphics Processing Unit (GPU) provided by Google Colab was utilized to train each model. The proposed machine learning model was developed and trained using Python, utilizing the Sklearn, Keras, Pandas, and NumPy libraries. For training, the models were trained for 30 epochs with a batch size of 32 and an initial learning rate of 0.001 (1e-3). The dataset was divided into training and testing sets, with 70% of the data allocated for training the model and the remaining 30% for testing its performance.

4.3.3.1 K-Nearest Neighbours Training

It works by finding the distance between a point and data with the selected specific number closest to the point and then votes for the most frequent label. Parameter `n_neighbors` is very important as it changes the accuracy rate varies. `n_neighbors` are several neighbors that can be used by default for neighbor queries. There is no specific method to choose the best value for it. The default value for it is 5. To determine a more optimal value for `n_neighbors`, you can test various values within a range (1 to 40) and evaluate the accuracy for each. After identifying the

accuracies corresponding to different `n_neighbors` values, the highest accuracy will be chosen. Let us determine the optimal value for the `n_neighbors` parameter by testing values from 1 to 20 and calculating the accuracy rate for each. Below are the results for each value within this range: The `k` (`n_neighbors`) value of 7 yields a marginally better accuracy rate than other values. Let us train the model using this parameter and observe the results.

```
Time to train knn on training dat: 0.1292433738708496
Accuracy - 99.40045992115637
```

4.3.3.2 Decision Tree Model Training

The decision tree classifier is a machine learning model that predicts a target variable's value based on multiple input features. It does this by recursively splitting the data based on the feature that provides the most information gain, creating a tree-like structure of decisions. The input features used to build the decision tree have been described in section 5.2. The decision tree will learn patterns in this data to predict the final status of the technical networks. The decision tree diagram visualizes the structure of the trained model. It shows the sequence of decisions made at each tree node, based on the most informative features. By following the branches of the tree, you can see how the model makes predictions for different combinations of input features.

- Decision Tree Model: Initialized with ``random_state=42``.
- Training: Model trained on the training dataset.

```
=====
Fitting Decision Tree Classifier
=====
Time to train RF on binary training dat: 0.08436155319213867
=====
Accuracy for binary Decision Tree is - 99.67969776609723
```

The model achieves an accuracy of 99.68%, surpassing the results obtained with original data.

4.3.3.3 Random Forest Model Training

This algorithm utilizes randomly selected sets of decision trees. It builds decision trees from randomly chosen data samples, makes predictions with each tree, and identifies the optimal solution through majority voting. To implement this algorithm, you need to import the Random Forest Classifier from the `sci-kit-learn` library, which provides free supervised and unsupervised machine learning algorithms for Python. The `n_estimators` parameter, which is optional and defaults to 100, specifies the number of trees in the forest. The Random Forest algorithm builds

multiple decision trees on random subsets of the training data. Each individual decision tree is trained on a different random subset of the features. This introduces randomness and diversity into the ensemble of decision trees. To make a prediction, the Random Forest takes the input features and passes them through each of the individual decision trees. The output of the Random Forest is the majority vote or average of the predictions made by the individual decision trees. The ensemble of multiple decision trees typically outperforms a single decision tree. The randomness introduced in the model-building process helps to reduce over-fitting. Random Forests can provide insights into the relative importance of each input feature in the prediction process.

The parameters in the Random Forest model are:

- `n_estimators`: The number of decision trees in the ensemble (100 in this case).
- `random_state`: A seed value to ensure reproducibility of the results (42 in this case).

```
=====
Fitting Random Forest Classifier
=====
Time to train RF on binary training dat: 0.45516014099121094
=====
Accuracy for binary RF is - 99.64273981603154
```

The model attains 99.64% accuracy, exceeding the performance achieved with the initial data.

4.3.3.4 Support Vector Machine Model Training

Support Vector Machine (SVM) is a fast and reliable classification algorithm that performs exceptionally well with limited data. It processes pairs of (x, y) coordinates to find the hyperplane that best separates the classes, forming the decision boundary. Items on one side of this boundary are classified as class A, while those on the other side are classified as class B. The algorithm uses a kernel parameter to specify the type of kernel function, which can be linear, poly, rbf, sigmoid, precomputed, or a callable function. The default kernel is rbf. In this case, a poly kernel with a degree of 1 and $C=3$ will be implemented. The code trains an SVM classifier, records the training time, evaluates its accuracy, generates a classification report, and saves the model to disk if it hasn't been saved already.

```
=====
Fitting SVM Classifier
=====
```

```
Time to train SVM on training dat: 277.83272194862366
Accuracy for binary SVM is - 98.7803876478318
```

4.3.3.5 Ensemble Model Training

Ensemble learning combines multiple classifiers to create a more potent model. Leveraging the strengths of different algorithms and using a voting system to make decisions can minimise weaknesses. The ML models, RF, KNN and SVM classifiers were trained using a hybrid approach. The results showed significant performance enhancements, indicating the effectiveness of the ensemble method, as depicted in Figure 4.2.

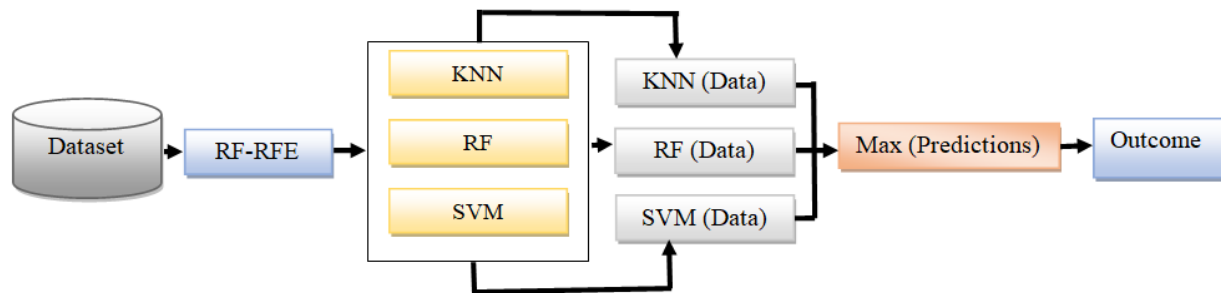


Figure 4.2: Staking Ensemble Model

Ensemble

Initialize Models: KNN, SVM, and Random Forest.

Train Models: Train each model on the training data (X_{train} , y_{train}).

Create Ensemble: Combine the trained models using a Voting Classifier.

Make Predictions: Use the ensemble to predict the test data (X_{test}).

Evaluate: Print accuracy and classification report.

Save Model: Save the ensemble model to disk if not already saved.

```
=====
Fitting Our Ensemble Method Classifier
=====
```

```
Time to train clf_voting on binary training dat: 281.23553228378296
=====
Accuracy for binary clf_voting is - 99.49080157687253
```

CHAPTER FIVE

RESULTS AND DISCUSSIONS

5.1 Introduction

Experiments were conducted in this thesis work with the Python programming language. NumPy, SciPy, Scikit-learn, Keras, Theano, and other libraries for machine learning, artificial intelligence, and computation are only a few available for Python, a potent, high-level, object-oriented language. The suggested models were constructed and trained using the Python ecosystem's Keras module, and TensorFlow was used for the actual execution. This section uses machine learning techniques to forecast intrusion in Mobile Ad Hoc Networks (MANET). The input dataset comprises cyber security data encompassing nine distinct network attacks observed in a commercial IP-based surveillance system and an IoT network. Before implementation in machine learning models, the data underwent pre-processing and formatting. Various machine learning algorithms were employed and assessed, such as KNN, Decision Trees, Random Forests, Support Vector Machines, and Ensembles. Hyperparameters for each algorithm were fine-tuned to enhance prediction performance. Predictive results were evaluated using a confusion matrix, revealing metrics like true positive, true negative, false positive, and false negative rates. Overall prediction accuracy, precision, recall, and F1-score were computed and compared across different algorithms. A variable importance test was also conducted to identify significant factors influencing intrusion predictions.

5.2. Model Evaluation and Performance Analysis

5.2.1 Evaluation of K-nearest Neighbours Model

Figure 5.1 provides a comprehensive overview of the evaluation of the Logistic Regression model. It includes detailed information on the testing results of the model's predictions.

```
Training Data Shape: (56821, 16)
Training Labels Shape: (56821,)
Testing Data Shape: (24352, 16)
Testing Label Shape: (24352,)
```

	precision	recall	f1-score	support
0	0.22	0.10	0.14	166
1	0.00	0.00	0.00	32
2	0.07	0.02	0.03	552
3	0.71	0.73	0.72	4881
4	0.31	0.07	0.12	518
5	0.89	0.99	0.94	11794
6	0.99	0.99	0.99	5871
7	0.12	0.01	0.01	505
8	0.00	0.00	0.00	33
accuracy			0.87	24352
macro avg	0.37	0.32	0.33	24352
weighted avg	0.82	0.87	0.84	24352

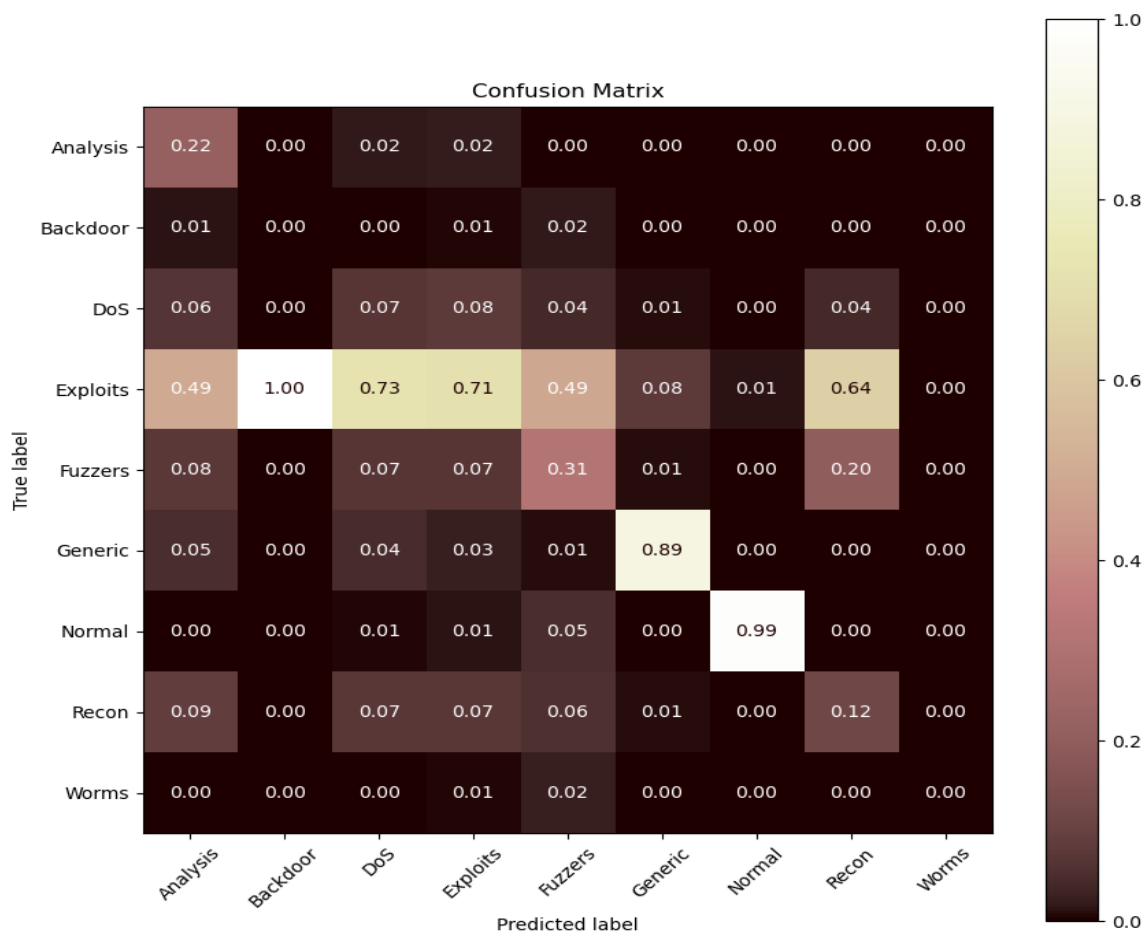


Figure 5.1: Classification Report and Confusion Matrix of KNN

5.2.2 Evaluation of Decision Tree Model

Model prediction testing results report,

	precision	recall	f1-score	support
0	0.58	0.41	0.48	166
1	0.00	0.00	0.00	32
2	0.48	0.20	0.28	552
3	0.77	0.79	0.78	4881
4	0.78	0.08	0.14	518
5	0.90	1.00	0.95	11794
6	1.00	0.99	0.99	5871
7	0.56	0.01	0.02	505
8	0.00	0.00	0.00	33
accuracy			0.89	24352
macro avg	0.56	0.39	0.40	24352
weighted avg	0.87	0.89	0.87	24352

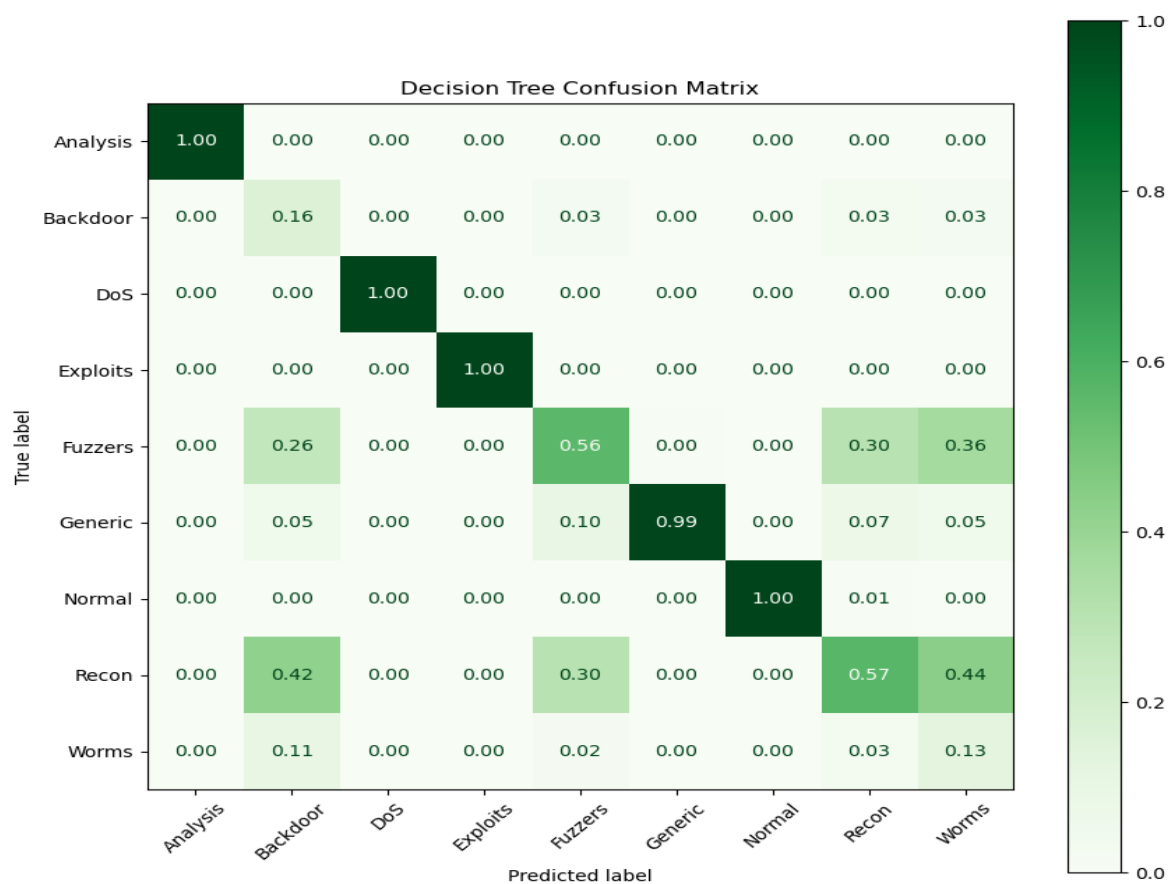


Figure 5.2: Classification Report and Confusion Matrix of Decision Tree Model

5.2.3 Evaluation of Random Forest Model

A Model Prediction Testing Result Report assesses a model's performance using key metrics, visualizations, and comparisons, providing insights and recommendations for improvement.

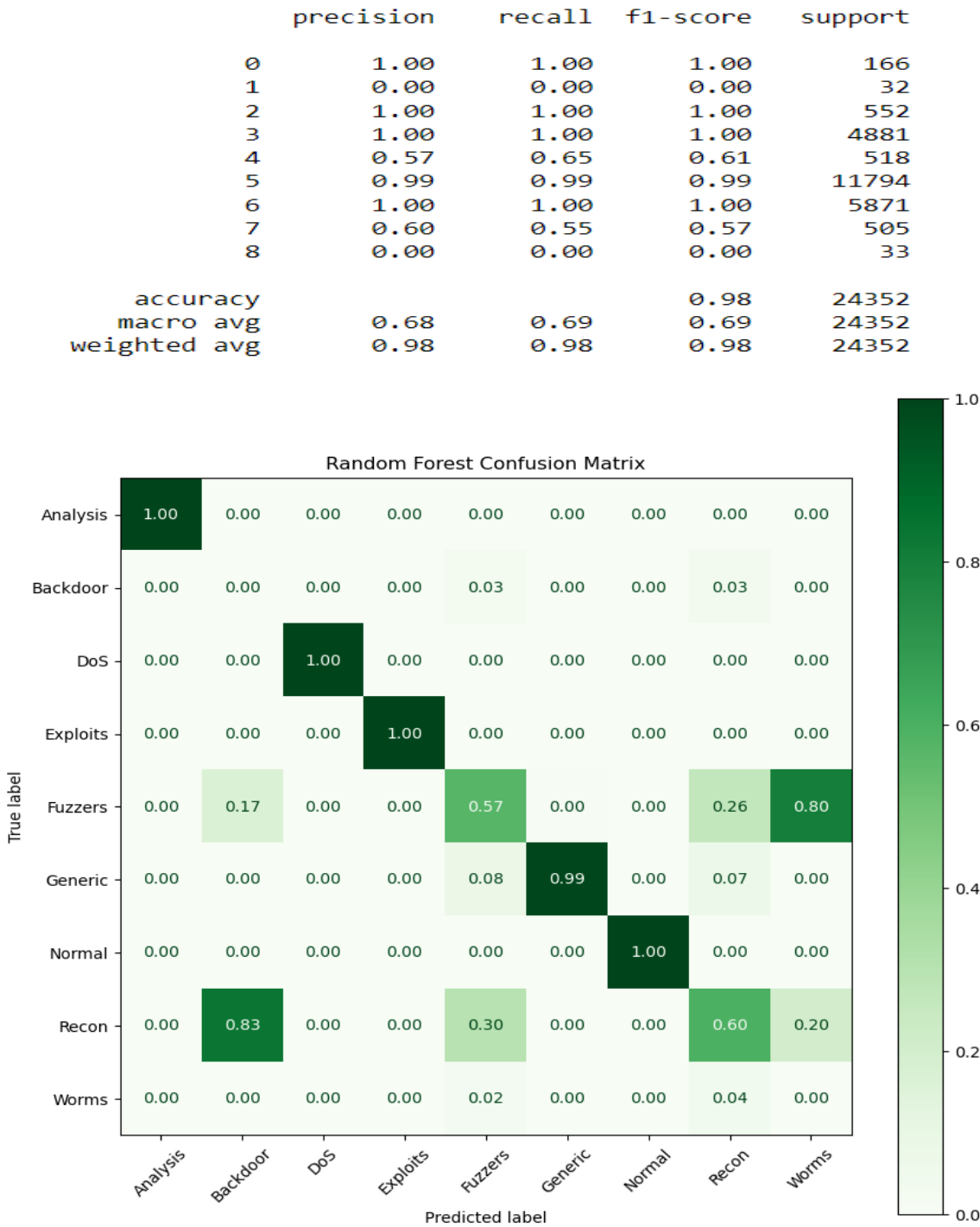


Figure 5.3: Classification Report and Confusion Matrix of Random Forest Model

5.2.4 Evaluation of the SVM Model

Model Prediction Testing Result Report evaluates a predictive model's performance using metrics like accuracy, precision, recall, and error rates. It includes a confusion matrix, sample predictions, and comparisons with baseline models to highlight improvements. Visualizations such as charts and graphs illustrate trends, while statistical analysis ensures reliability. The report also provides recommendations for enhancing model performance, making it a key tool for assessment and decision-making.

	precision	recall	f1-score	support
0	0.00	0.00	0.00	166
1	0.00	0.00	0.00	32
2	0.00	0.00	0.00	552
3	0.69	0.65	0.67	4881
4	0.00	0.00	0.00	518
5	0.87	1.00	0.93	11794
6	0.91	0.96	0.93	5871
7	0.00	0.00	0.00	505
8	0.00	0.00	0.00	33
accuracy			0.85	24352
macro avg	0.27	0.29	0.28	24352
weighted avg	0.78	0.85	0.81	24352

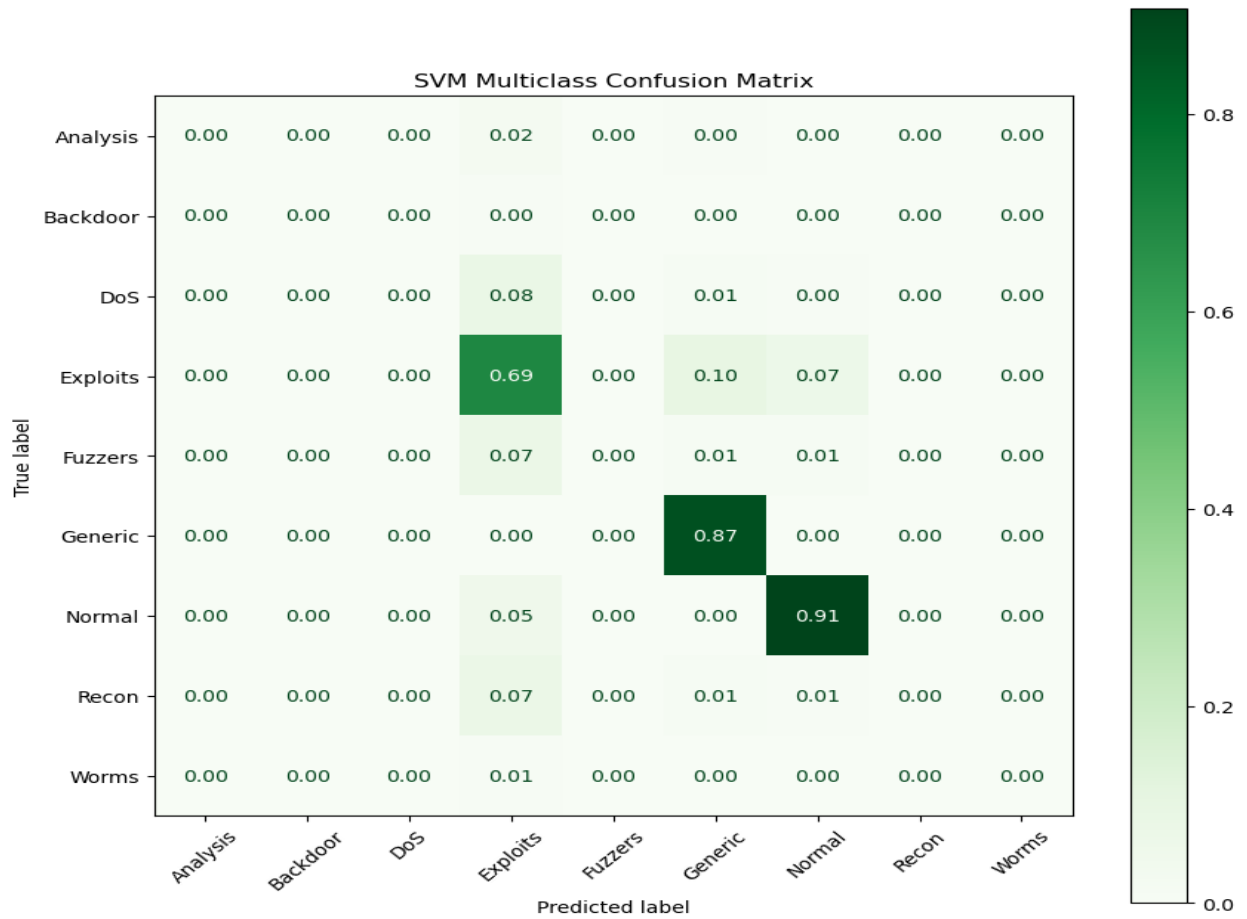


Figure 5.4: Classification Report and Confusion Matrix of SVM Model

5.2.5 Evaluation of Ensemble Model

Model prediction testing result reports,

	precision	recall	f1-score	support
0	1.00	1.00	1.00	166
1	0.12	0.06	0.08	32
2	1.00	1.00	1.00	552
3	1.00	1.00	1.00	4881
4	0.56	0.56	0.56	518
5	0.99	0.99	0.99	11794
6	1.00	1.00	1.00	5871
7	0.56	0.63	0.59	505
8	0.00	0.00	0.00	33
accuracy			0.98	24352
macro avg	0.69	0.69	0.69	24352
weighted avg	0.98	0.98	0.98	24352

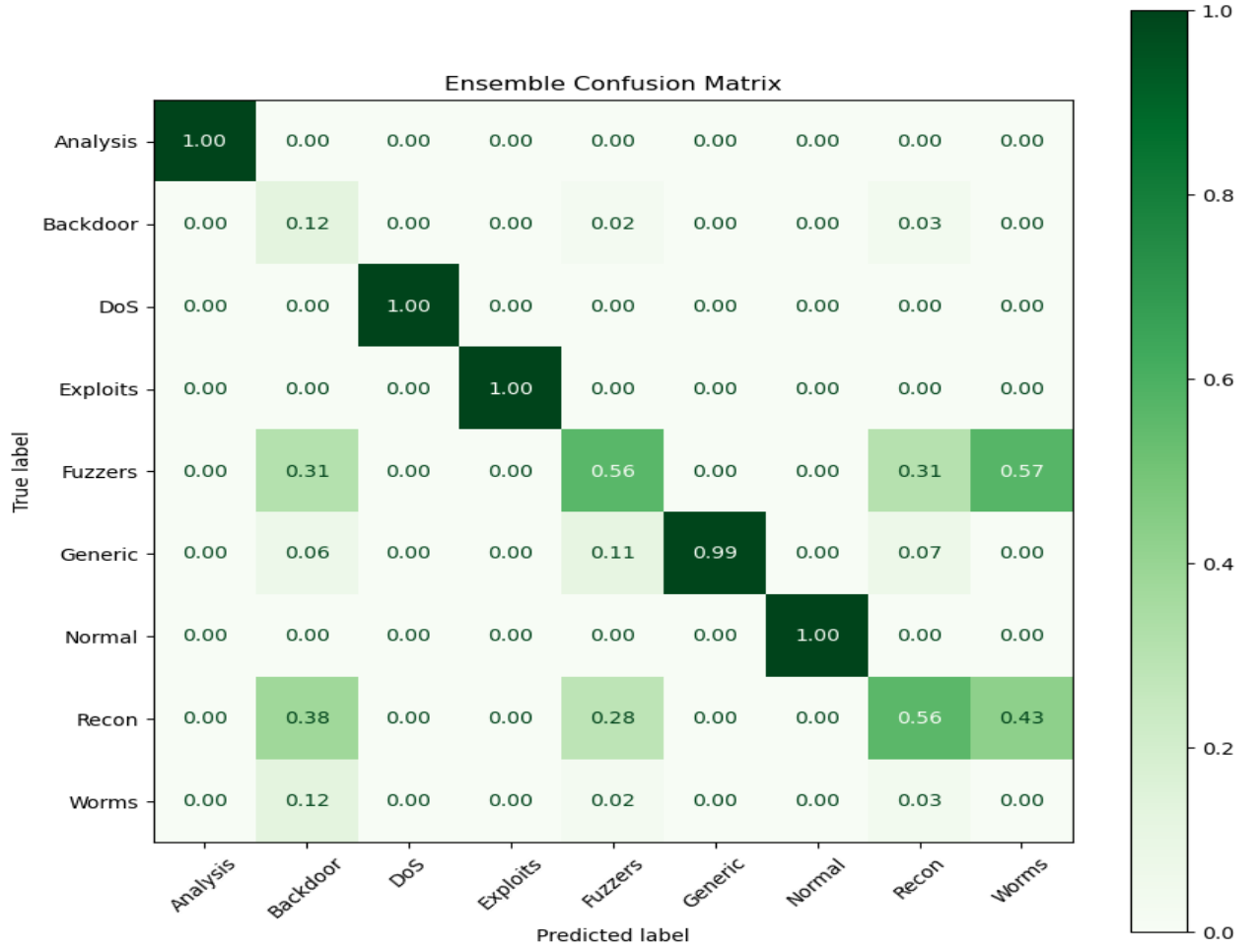


Figure 5.5: Classification Report and Confusion Matrix of Ensemble Model

The proposed approach's performance was evaluated using the pre-processed dataset. Experimental results for this approach are shown in Table 5.1.

Table 5.1: Dataset with features selected by RF-RFE (16 features).

Classifier	Accuracy	Precision	Recall	F1-Score
Random forest	97.73%	98.01%	98.02%	98.01%
KNN	86.50%	82.31%	87.41%	84.78%
SVM	84.60%	78.11%	85.40%	81.59%
Decision tree	97.58%	98.24%	98.12%	98.17%
Ensemble	98.53%	98.79%	98.78%	98.78%

The hybrid stacking ensemble approach chose to employ RF. The results presented in Table 5.1 show that the suggested strategy, which makes use of RF-RFE and the stacking-based ensemble method, outperforms individual machine-learning algorithms by a large margin. This algorithm

outperforms both RF and other techniques, achieving over 98.53% accuracy. The ROC curve for the suggested ensemble model using the UNSW-NB15 dataset is shown in Figure 5.6. It shows that, for the majority of classes, the True Positive Rate (TPR) is high and close to 1, while the False Positive Rate (FPR) stays low, as intended. It is therefore possible to conclude that the suggested method successfully detects Denial of Service (DoS) assaults based on the data presented in Table 5.1 and the ROC curve. The method's high capacity to differentiate between these classes is indicated by the ROC curve, which also reveals that the area under the curve (AUC) is almost 1 for the majority of classes. Nonetheless, the AUC for classes 1 (Backdoor) and 8 (Worms) has significantly decreased. The imbalance in the dataset could explain this fall, the smaller sample sizes for these minority classes, and the possibility of losing significant features as a result of dimensionality reduction employing RF-RFE. Despite this, the suggested approach performs better overall than the separate methods.

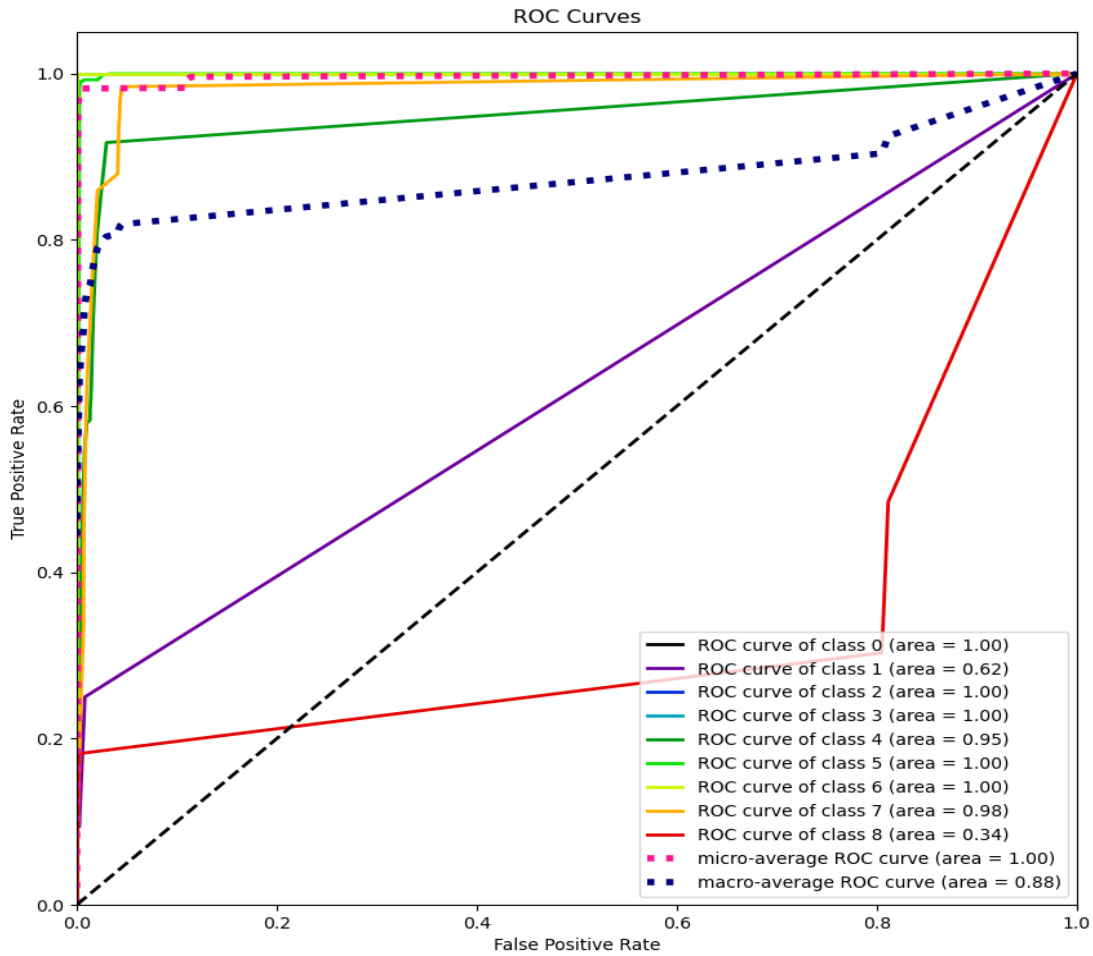


Figure 5.6: Receiver Operating Characteristic (ROC) curve for the dataset

5.3 Comparative Analysis with Recent Studies

The proposed stacking ensemble model, which integrates Random Forest, KNN, SVM, and Decision Trees, achieved an outstanding accuracy of 98.53% on the UNSW-NB15 dataset. This performance is superior to several recent studies that have applied machine learning and deep learning techniques for DoS attack detection. For instance, Dora and Lakshmi (2022) implemented a CNN with LSTM for DDoS detection, achieving an accuracy of 96.37%. While their approach showed promise, it required substantial computational resources and was less suitable for real-time applications. Similarly, Prathyusha and Kannayaram (2021) utilized Artificial Immune Systems (AIS) for cloud-based DoS detection, obtaining a 96.7% accuracy. Although this method demonstrated reasonable success, it lacked scalability and required significant tuning efforts.

Another relevant study by Bhardwaj et al. (2020) applied a Deep Neural Network (DNN) with an Autoencoder (AE), reaching an accuracy of 98.43%. However, this method faced computational challenges despite its high accuracy, making it less practical for deployment in resource-constrained environments. Mukhametzhanov et al. (2019) developed a Neural Network-based single-layer perceptron model that achieved 97.94% accuracy, but its lack of scalability limited its effectiveness in dynamic network conditions.

Verma et al. (2020) proposed a hybrid approach combining Random Forest with Mean Absolute Deviation (MAD) thresholding, obtaining an accuracy of 98.23%. Although this method was competitive, it suffered from the complexity of threshold tuning, which can be challenging to optimize in practical scenarios. In contrast, the proposed ensemble model achieves superior accuracy and ensures computational efficiency, making it an optimal choice for real-world intrusion attack detection. The study highlights how this approach can seamlessly integrate into intrusion detection systems for critical applications, such as securing IoT networks, cloud infrastructures, and innovative city implementations. This study's findings emphasize the importance of ensemble learning in cybersecurity. By combining multiple classifiers, the proposed model enhances detection reliability, reduces computational costs, and ensures real-time applicability in various security-sensitive environments.

5.4 Summary of Findings

The study's findings indicate that the proposed model significantly outperforms traditional machine learning models such as KNN, Decision Trees (DT), Random Forest (RF), and Support

Vector Machines (SVM) in terms of key performance metrics. Specifically, the model achieved an impressive F1 score of 98.78%, a recall of 98.78%, a precision of 98.79%, and a mean accuracy of 98.53%. These results illustrate the effectiveness of ensemble learning in improving detection accuracy. Unlike single-model approaches, which often suffer from biases and overfitting issues, the ensemble method leverages the strengths of multiple algorithms. Combining different classifiers enhances decision-making and reduces the risk of misclassification. The study further highlights that the model's ability to detect DoS attacks using the UNSW-NB15 dataset is superior to other existing models, reinforcing its reliability in real-world cybersecurity applications. Moreover, these performance gains suggest that the ensemble model is a robust solution for intrusion detection systems (IDS). Since cyber threats continue to evolve, a strong IDS must be adaptive and capable of handling new attack variations. The ensemble method proposed in this study meets these requirements, making it a promising approach for modern cybersecurity defenses.

One key advantage of the proposed ensemble approach is its efficiency in real-world applications, particularly in environments with limited computational resources. Unlike deep learning models that require extensive datasets and high-end GPUs, the ensemble method achieves superior performance while maintaining lower computational costs.

This advantage is particularly crucial for edge computing environments such as the Internet of Things (IoT) and Mobile Ad Hoc Networks (MANETs), where computational power and memory resources are constrained. Deploying deep learning models in such environments is impractical due to the high energy consumption and latency in processing large datasets. In contrast, the ensemble model effectively balances computational efficiency and detection accuracy, making it an ideal choice for real-time cyber threat mitigation. Another significant benefit of the ensemble method is its improved generalization capability. Single machine learning models often struggle with overfitting, performing well on training data but failing to maintain accuracy when encountering new data. The ensemble approach mitigates this issue by integrating multiple models, ensuring more robust performance across different datasets. Additionally, this approach enhances scalability, making it suitable for dynamic network environments where cyber threats continuously evolve. Furthermore, the ensemble model reduces the dependency on extensive labelled datasets, which is essential for deep learning models. Annotating large-scale datasets for cybersecurity applications is both time-consuming and expensive. The ensemble

approach effectively addresses this challenge by leveraging a diverse set of classifiers, thereby minimizing the need for exhaustive labelled data while achieving high accuracy.

This study introduces key contributions in research design and data analysis, including the development of an optimized feature selection strategy (RF-RFE) to enhance detection efficiency, the implementation of an ensemble-based model that outperforms individual classifiers in detecting intrusions, a refined experimental framework integrating hyperparameter tuning, cross-validation, and performance benchmarking for robust model evaluation, and the integration of domain expertise from MANET security professionals to validate the intrusion detection approach. These refinements enhance the research methodology's clarity, structure, and rigor, ensuring a well-validated and scalable intrusion detection system for MANET environments.

- Development of an optimized feature selection strategy (RF-RFE) to enhance detection efficiency.
- Implementation of an ensemble-based model that outperforms individual classifiers in detecting intrusions.
- Refined experimental framework integrating hyper-parameter tuning, cross-validation, and performance benchmarking for robust model evaluation.
- Integration of domain expertise from MANET security professionals to validate the intrusion detection approach.

5.5 Discussion

This study presents an advanced intrusion attack detection approach that leverages machine learning classifiers and model optimization strategies. Traditional intrusion detection systems often rely on outdated datasets and rule-based methods, limiting their effectiveness against evolving cyber threats. As cyberattacks become more sophisticated, the need for intelligent and adaptive detection mechanisms has become crucial. This research focuses on developing a more robust and efficient detection mechanism utilizing state-of-the-art machine learning techniques to enhance security in modern network environments.

The core of the methodology involves a diverse set of supervised learning algorithms, including K-nearest neighbours (KNN), Support Vector Machines (SVM), Decision Trees (DT), and Random Forests (RF). Each classifier possesses unique strengths that contribute to classifying network traffic into standard or attack categories. KNN helps detect anomalies by measuring the

similarity between data points, while SVM provides high accuracy in distinguishing between normal and malicious activities. Decision Trees offer interpretability and efficient classification, whereas Random Forests enhance reliability by combining multiple decision trees to reduce variance and overfitting.

The study integrates ensemble techniques to enhance predictive accuracy, combining multiple models to generate more reliable detection outcomes. This approach ensures that weaknesses of individual models are compensated for, leading to improved classification results. Moreover, the approach undergoes rigorous hyperparameter tuning, optimising classifier settings to improve performance. Cross-validation ensures the model generalizes well across different datasets and does not overfit specific data patterns. These optimizations help improve the detection rate of cyber threats while minimizing false alarms. A well-defined data utilization framework is crucial for achieving high detection accuracy and ensuring machine learning models learn from relevant, high-quality data. This framework includes key steps such as data collection, pre-processing, feature extraction, feature selection, and integration with machine learning models. Data collection ensures up-to-date, high-quality datasets representing modern cyber threats. Pre-processing involves handling missing values, normalizing data, and removing irrelevant attributes that do not contribute to intrusion detection. Without these pre-processing steps, noise in the data could degrade model performance and lead to misleading results.

Feature extraction is another critical step in the data utilization framework. It identifies the most informative features that improve model accuracy, such as network traffic volume, connection duration, and packet size distribution. These features help the classifiers distinguish between normal and malicious traffic patterns. Feature selection refines the dataset by retaining only the most critical attributes for detecting intrusion attacks. This step improves computational efficiency by reducing the dimensionality of the dataset while maintaining high predictive performance. Finally, the cleaned and processed data is integrated into classifiers for training and evaluation, ensuring that models receive only the most relevant information for attack detection. This research distinguishes itself by moving away from outdated datasets, such as NSL-KDD, which fail to capture modern attack patterns (Issa & Albayrak, 2023). Instead, the study advocates using contemporary datasets that reflect the latest attack methodologies. Including up-to-date data significantly improves the system's capability to detect novel and sophisticated intrusion attacks, making intrusion detection more effective in real-world scenarios. This

transition is crucial because outdated datasets often lack representations of emerging threats, making it difficult for models to adapt to evolving cyberattack strategies.

Key performance metrics, including accuracy, precision, recall, and computational efficiency, are used to assess the effectiveness of the proposed methodology. Accuracy measures the overall correctness of the model by comparing its predictions to actual labels. Precision evaluates how many detected attacks were actual threats, helping to assess the model's ability to minimize false positives. Recall determines the model's ability to identify actual attacks without missing them, ensuring that false negatives are minimized. Computational efficiency measures the model's ability to process real-time network traffic, an essential factor for deploying intrusion detection systems in live environments. A comparative analysis is performed against existing detection approaches to highlight improvements in detection rates, reduced false positives, and better computational efficiency. The proposed model's effectiveness is demonstrated by comparing its results with traditional rule-based and outdated machine-learning approaches. The evaluation provides insight into how well the proposed system adapts to modern cyber threats and ensures robust performance across diverse datasets.

CHAPTER SIX

CONCLUSION AND RECOMMENDATIONS

6.1. Conclusion

This study comprehensively explored the application of machine learning-based intrusion detection systems (IDS) for securing Mobile Ad Hoc Networks (MANETs). MANETs, characterized by their dynamic topology, decentralized nature, and resource constraints, are highly susceptible to various cyber threats, including denial-of-service (DoS) attacks, black hole attacks, and Sybil attacks. Traditional security mechanisms, such as encryption and authentication, are often insufficient due to these networks' highly adaptive and distributed nature. This study aimed to address these challenges by developing a robust, adaptive, and efficient IDS that can effectively detect and mitigate network intrusions in MANET environments.

To achieve this, the research successfully met its key objectives, starting with identifying and analysing common security threats in MANETs. A detailed study of network vulnerabilities and attack patterns provided the foundation for selecting appropriate machine learning models capable of classifying and detecting intrusions. The study leveraged the UNSW-NB15 dataset, a widely recognized dataset for cybersecurity research, to train and evaluate multiple machine learning classifiers. The selected models included Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), and an ensemble-based model that integrates multiple classifiers to enhance detection accuracy.

The ensemble approach demonstrated superior performance to individual classifiers through rigorous experimental analysis. The final evaluation metrics showed that the ensemble model achieved an accuracy of 98.53%, a precision of 98.79%, a recall of 98.78%, and an F1-score of 98.78%. These results confirm that ensemble learning enhances intrusion detection accuracy by combining the strengths of multiple classifiers, reducing misclassification rates, and improving overall system reliability. Furthermore, the study optimised computational efficiency without compromising detection effectiveness by applying feature selection techniques such as Recursive Feature Elimination with Random Forest (RF-RFE).

The findings of this study contribute to the ongoing efforts to improve MANET security by demonstrating that machine learning-based IDS can significantly enhance intrusion detection and

classification. The proposed model minimizes false positives while ensuring high recall rates, making it a promising solution for real-world deployments. Future research should focus on integrating real-time adaptability, developing lightweight models suitable for resource-constrained MANET nodes, and testing the IDS in live network environments to validate its practical applicability.

6.2. Contributions of the Thesis

This study introduces several significant contributions to the field of MANET security. Developing an optimized feature selection strategy (RF-RFE) enhances detection efficiency while reducing computational overhead. Implementing an ensemble-based model demonstrates superior intrusion detection and classification performance compared to individual classifiers. A comparative analysis of multiple machine learning models provides empirical insights into their effectiveness for intrusion detection. The study also integrates hyper-parameter tuning and cross-validation techniques, ensuring a robust model evaluation framework. Furthermore, the involvement of domain experts in MANET security validates the proposed IDS, ensuring its alignment with real-world security challenges.

- Developed an optimized feature selection strategy (RF-RFE) to enhance detection efficiency while reducing computational overhead.
- Implemented an ensemble-based model that outperforms individual classifiers in detecting and classifying intrusions in MANET environments.
- Conducted a comparative analysis of multiple machine learning models, providing empirical evidence on their effectiveness for intrusion detection.
- Integrated hyper-parameter tuning and cross-validation techniques to ensure robust model evaluation.

6.3. Recommendations for Future Work

Future research should enhance real-time adaptability by implementing online or incremental machine learning models that dynamically adjust to evolving intrusion patterns. Integrating the IDS with live MANET deployments, such as military and emergency response networks, will provide practical validation of its applicability. Developing energy-efficient IDS models is essential to optimize performance on resource-constrained MANET nodes while minimizing power consumption. Advanced threat detection techniques, including deep learning-based approaches, should be explored to improve detection accuracy, particularly for novel attack

patterns. To further enhance the effectiveness and scalability of the proposed IDS, future research should focus on the following areas:

- **Real-time Adaptability:** Implementing online or incremental machine learning models to adapt to evolving intrusion patterns in real-time.
- **Integration with Live MANET Deployments:** Conducting real-world IDS testing in live MANET environments, such as military or emergency response networks, to evaluate its practical applicability.
- **Energy-Efficient IDS Models:** Developing lightweight IDS models optimized for resource-constrained MANET nodes, ensuring minimal impact on network performance.
- **Advanced Threat Detection Techniques:** Incorporating deep learning-based models to improve detection accuracy, especially for novel attack patterns.
- **Policy and Framework Integration:** Collaborating with national cybersecurity institutions such as INSA, Ethio Telecom, and the Ministry of Defense to align the IDS with existing cybersecurity policies and frameworks.

REFERENCES

- Abbas, T., Qamar, F., Hindia, M. H. D. N., Hassan, R., Ahmed, I., & Aslam, M. I. (2020). Performance Analysis of Ad Hoc on-Demand Distance Vector Routing Protocol for MANET. *2020 IEEE Student Conference on Research and Development, SCORED 2020, September*, 194–199. <https://doi.org/10.1109/SCORED50371.2020.9250989>
- Abbood, Z. A., Atilla, D. Ç., & Aydin, Ç. (2023). Intrusion detection system through deep learning in routing manet networks. *Intelligent Automation and Soft Computing*, 37(1), 269–281. <https://doi.org/10.32604/iasc.2023.035276>
- Abdan, M., & Seno, S. A. H. (2022). Machine Learning Methods for Intrusive Detection of Wormhole Attack in Mobile Ad Hoc Network (MANET). *Wireless Communications and Mobile Computing*, 2022, 0–21. <https://doi.org/10.1155/2022/2375702>
- Abdullah, A. M. (2021). *Mobile Ad hoc Networks: A Survey of Existing Mobility Models and Routing Protocols* Call for papers View project Application of Petri Nets in Computer Networks View project Mobile Ad hoc Networks: A Survey of Existing Mobility Models and Routing Protocols (Issue August). <https://www.researchgate.net/publication/354219470>
- AbualSaud, K., Elfouly, T. M., Khattab, T., Yaacoub, E., Ismail, L. S., Ahmed, M. H., & Guizani, M. (2019). A Survey on Mobile Crowd-Sensing and Its Applications in the IoT Era. *IEEE Access*, 7, 3855–3881. <https://doi.org/10.1109/ACCESS.2018.2885918>
- Abubakar, R., Aldegheishem, A., Faran Majeed, M., Mehmood, A., Maryam, H., Ali Alrajeh, N., Maple, C., & Jawad, M. (2020). An Effective Mechanism to Mitigate Real-Time DDoS Attack. *IEEE Access*, 8, 126215–126227. <https://doi.org/10.1109/ACCESS.2020.2995820>
- Agrawal, R., Faujdar, N., Romero, C. A. T., Sharma, O., Abdulsahib, G. M., Khalaf, O. I., Mansoor, R. F., & Ghoneim, O. A. (2023). Classification and comparison of ad hoc networks: A review. *Egyptian Informatics Journal*, 24(1), 1–25. <https://doi.org/10.1016/j.eij.2022.10.004>
- Al Sawafi, Y., Touzene, A., Day, K., & Alzeidi, N. (2020). Mobile Crowd Sensing RPL-based routing protocol for smart city. *International Journal of Computer Networks and Communications*, 12(2), 49–68. <https://doi.org/10.5121/ijcnc.2020.12203>
- Alameri, I. A. (2018). *ETASR_1810.pdf*. 8(1), 2604–2608.
- Alauthman, M., Aslam, N., Al-kasassbeh, M., Khan, S., Al-Qerem, A., & Raymond Choo, K. K. (2020). An efficient reinforcement learning-based Botnet detection approach. *Journal of*

- Network and Computer Applications*, 150, 102479.
<https://doi.org/10.1016/j.jnca.2019.102479>
- Alghazzawi, D., Bamasaq, O., Bhatia, S., Kumar, A., Dadheech, P., & Albeshri, A. (2021). Congestion Control in Cognitive IoT-Based WSN Network for Smart Agriculture. *IEEE Access*, 9, 151401–151420. <https://doi.org/10.1109/ACCESS.2021.3124791>
- Ali, M., Haque, M. ul, Durad, M. H., Usman, A., Mohsin, S. M., Mujlid, H., & Maple, C. (2023). Effective network intrusion detection using stacking-based ensemble approach. *International Journal of Information Security*, 22(6), 1781–1798.
<https://doi.org/10.1007/s10207-023-00718-7>
- Alo, R. U., Stanly, N. I., & Onwe, N. F. (2018). Mobile Ad Hoc Network (MANET): Applications, Benefits and Performance Issues in a Global Positioning System. *International Research Journal of ...*, 983–987.
https://www.academia.edu/download/58431429/manuel_paper.pdf
- Alshammari, D., & Al-Shehri, S. (2019). MANETs in Military Communications Design Concepts and Challenges. *International Journal of Innovation Engineering and Science Research*, 3(2), 2581–4591. www.ijiesr.com
- Anjum, M., & Shreedhara, K. S. (2019). Performance Analysis of Semi-Supervised Machine Learning Approach for DDoS Detection. *International Journal of Innovative Research in Technology*, 6(2), 144–147.
- Anjum, S. S., Noor, R. M., & Anisi, M. H. (2017). Review on MANET Based Communication for Search and Rescue Operations. *Wireless Personal Communications*, 94(1), 31–52.
<https://doi.org/10.1007/s11277-015-3155-y>
- Arden, N., Bensah, L., Agyekum, B., Ankora, C., Tietaa, G., & Mawuena, A. (2021). Black Hole Attack in Mobile Ad Hoc Networks: Challenges and Directions. *International Journal of Computer Applications*, 183(29), 1–7. <https://doi.org/10.5120/ijca2021921672>
- Bagyalakshmi, C. (2020). DDoS Attack Classification on Cloud Environment Using Machine Learning Techniques with Different Feature Selection Methods. *International Journal of Advanced Trends in Computer Science and Engineering*, 9(5), 7301–7308.
<https://doi.org/10.30534/ijatcse/2020/60952020>
- Basil, N., Ahammad, S. H., & Elsayed, E. E. (2024). Enhancing wireless subscriber performance through AODV routing protocol in simulated mobile Ad-hoc networks. 3(1), 16–26.

- Bhardwaj, A., & El-ocla, H. (2020). *Multipath Routing Protocol Using Genetic Algorithm in Mobile Ad Hoc Networks*. 8. <https://doi.org/10.1109/ACCESS.2020.3027043>
- Bhardwaj, A., Mangat, V., & Vig, R. (2020). Hyperband tuned deep neural network with well posed stacked sparse autoencoder for detection of ddos attacks in cloud. *IEEE Access*, 8, 181916–181929. <https://doi.org/10.1109/ACCESS.2020.3028690>
- Bhatti, D. S., & Saleem, S. (2020). Ephemeral secrets: Multi-party secret key acquisition for secure IEEE 802.11 mobile ad hoc communication. *IEEE Access*, 8, 24242–24257. <https://doi.org/10.1109/ACCESS.2020.2970147>
- Biswas, R., Kim, S., & Wu, J. (2021). Sampling Rate Distribution for Flow Monitoring and DDoS Detection in Datacenter. *IEEE Transactions on Information Forensics and Security*, 16, 2524–2534. <https://doi.org/10.1109/TIFS.2021.3054522>
- Chen, P., Xie, Z., Fang, Y., Chen, Z., Mumtaz, S., & Rodrigues, J. J. P. C. (2020). Physical-Layer Network Coding: An Efficient Technique for Wireless Communications. *IEEE Network*, 34(2), 270–276. <https://doi.org/10.1109/MNET.001.1900289>
- Chen, Q., Meng, Z., Liu, X., Jin, Q., & Su, R. (2018). Decision variants for the automatic determination of optimal feature subset in RF-RFE. *Genes*, 9(6). <https://doi.org/10.3390/genes9060301>
- Danilchenko, K., Azoulay, R., Reches, S., & Haddad, Y. (2023). Deep Learning for MANET Routing. *IEEE Transactions on Machine Learning in Communications and Networking*, 1(October), 412–424. <https://doi.org/10.1109/tmlcn.2023.3324280>
- Das, S., Mahfouz, A. M., Venugopal, D., & Shiva, S. (2019). DDoS Intrusion Detection Through Machine Learning Ensemble. *Proceedings - Companion of the 19th IEEE International Conference on Software Quality, Reliability and Security, QRS-C 2019*, 471–477. <https://doi.org/10.1109/QRS-C.2019.00090>
- Dora, V. R. S., & Lakshmi, V. N. (2022). Optimal feature selection with CNN-feature learning for DDoS attack detection using meta-heuristic-based LSTM. *International Journal of Intelligent Robotics and Applications*, 6(2), 323–349. <https://doi.org/10.1007/s41315-022-00224-4>
- Eltanbouly, S., Bashendy, M., Alnaimi, N., Chkirbene, Z., & Erbad, A. (2020). Machine Learning Techniques for Network Anomaly Detection: A Survey. *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies, ICIoT 2020*, 156–162.

<https://doi.org/10.1109/ICIoT48696.2020.9089465>

- Fathima, A., Devi, G. S., & Faizaanuddin, M. (2023). Improving distributed denial of service attack detection using supervised machine learning. *Measurement: Sensors*, 30(August), 100911. <https://doi.org/10.1016/j.measen.2023.100911>
- Guo, Y., Li, Z., Lv, R., & Yang, Z. (2022). Evaluating the impact of aggregation and RTS/CTS schemes on IEEE 802.11 based Linear Wireless Ad-Hoc Networks. *Engineering Reports*, 4(11), 1–21. <https://doi.org/10.1002/eng2.12516>
- Gupta, C., Singh, L., & Tiwari, R. (2022). Wormhole attack detection techniques in ad-hoc network: A systematic review. *Open Computer Science*, 12(1), 260–288. <https://doi.org/10.1515/comp-2022-0245>
- Hmeed, A. R., Hammad, J. A., & Obaid, A. J. (2023). *INTELLIGENT SYSTEMS AND APPLICATIONS IN Enhanced Quality of Service (QoS) for MANET Routing Protocol Using a Distance Routing Effect Algorithm for Mobility (DREAM)*. 11, 409–417.
- Idhammad, M., Afdel, K., & Belouch, M. (2018). Semi-supervised machine learning approach for DDoS detection. *Applied Intelligence*, 48(10), 3193–3208. <https://doi.org/10.1007/s10489-018-1141-2>
- IEEE Standards Association. (2010). IEEE Standard for Information technology Telecommunications and Information exchange between IEEE Standard for Information Technology Telecommunications and information exchange between systems Local and metropolitan area networks Specific Requirement.Par. In *IEEE Standards Association* (Vol. 2007, Issue June). <http://standards.ieee.org/regauth/faqs.html>. Copyright
- Igbe, O., Ajayi, O., & Saadawi, T. (2017). Denial of service attack detection using dendritic cell algorithm. *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference, UEMCON 2017, 2018-Janua*(November), 294–299. <https://doi.org/10.1109/UEMCON.2017.8249054>
- Islam, M. N. U., Fahmin, A., Hossain, M. S., & Atiquzzaman, M. (2021). Denial-of-Service Attacks on Wireless Sensor Network and Defense Techniques. In *Wireless Personal Communications* (Vol. 116, Issue 3). Springer US. <https://doi.org/10.1007/s11277-020-07776-3>
- Issa, A. S. A., & Albayrak, Z. (2023). DDoS Attack Intrusion Detection System Based on Hybridization of CNN and LSTM. *Acta Polytechnica Hungarica*, 20(2), 105–123.

<https://doi.org/10.12700/APH.20.2.2023.2.6>

- Jemili, F., Meddeb, R., & Korbaa, O. (2023). Intrusion detection based on ensemble learning for big data classification. *Cluster Computing*, 1–37. <https://doi.org/10.1007/s10586-023-04168-7>
- Joardar, S., Sinhababu, N., Dey, S., & Choudhury, P. (2023). Mitigating DoS Attack in MANETs Considering Node Reputation with AI. In *Journal of Network and Systems Management* (Vol. 31, Issue 3). Springer US. <https://doi.org/10.1007/s10922-023-09742-3>
- Kanellopoulos, D., & Cuomo, F. (2021). Recent developments on mobile ad-hoc networks and vehicular ad-hoc networks. *Electronics (Switzerland)*, 10(4), 1–3. <https://doi.org/10.3390/electronics10040364>
- Kemmerer, R. A., & Vigna, G. (2002). Intrusion detection: A brief history and overview. *Computer*, 35(SUPPL.), 27–30. <https://doi.org/10.1109/mc.2002.1012428>
- Khudayer, B. H., & Anbar, M. (2020). *Efficient Route Discovery and Link Failure Detection Mechanisms for Source Routing Protocol in Mobile Ad-Hoc Networks*. 8, 24019–24032.
- KONJIT, D. (2006). INTRUSION DETECTION SYSTEM FOR MOBILE AD-HOC NETWORKS (MANETs). <https://Etd.Aau.Edu.et/Items/Af36f879-Cc0f-4a16-9918-Fccc817dc8e8>, 44(2), 8–10.
- Krishnamoorthy, V. K., Izonin, I., Subramanian, S., Shandilya, S. K., Velayutham, S., Munichamy, T. R., & Havryliuk, M. (2023). *Energy Saving Optimization Technique-Based Routing Protocol in Mobile Ad-Hoc Network with IoT Environment*.
- Kurumbanshi, S. (2023). *An efficient framework for application layer DDoS attack detection and mitigation for Mobile Ad Hoc Network*.
- Kushwah, R., & Garg, R. (2024). Machine Learning Algorithms for Attack and Anomaly Detection in IoT. *Lecture Notes in Electrical Engineering*, 1075 LNEE(MI), 291–303. https://doi.org/10.1007/978-981-99-5091-1_21
- Lata, S., & Singh, D. (2022). Intrusion detection system in cloud environment: Literature survey & future research directions. *International Journal of Information Management Data Insights*, 2(2), 100134. <https://doi.org/10.1016/j.jjime.2022.100134>
- Leevy, J. L., & Khoshgoftaar, T. M. (2020). A survey and analysis of intrusion detection models based on CSE-CIC-IDS2018 Big Data. *Journal of Big Data*, 7(1). <https://doi.org/10.1186/s40537-020-00382-x>

- Ma, L., Chai, Y., Cui, L., Ma, D., Fu, Y., & Xiao, A. (2020). A Deep Learning-Based DDoS Detection Framework for Internet of Things. *IEEE International Conference on Communications, 2020-June*. <https://doi.org/10.1109/ICC40277.2020.9148944>
- Mahmodi, A., Derakhsh, A. M., Daneshjoo, P., & Delara, C. (2018). Using Genetic Algorithm to Improve Bernoulli Naïve Bayes Algorithm in Order to Detect DDoS Attacks in Cloud Computing Platform. *International Journal of Science and Engineering Investigations*, 7(January). <https://www.researchgate.net/publication/323961156>
- Marteau, P. F. (2019). Sequence Covering for Efficient Host-Based Intrusion Detection. *IEEE Transactions on Information Forensics and Security*, 14(4), 994–1006. <https://doi.org/10.1109/TIFS.2018.2868614>
- Mast, N., Khan, S., Uddin, M. I., Ghadi, Y. Y., Alkahtani, H. K., & Mostafa, S. M. (2023). A Cross-Layer Solution for Contention Control to Enhance TCP Performance in Wireless Ad-Hoc Networks. *IEEE Access*, 11(February), 24875–24893. <https://doi.org/10.1109/ACCESS.2023.3244888>
- Mayadunna, H., De Silva, S. L., Wedage, I., Pabasara, S., Rupasinghe, L., Liyanapathirana, C., Kesavan, K., Nawarathna, C., & Sampath, K. K. (2017). Improving trusted routing by identifying malicious nodes in a MANET using reinforcement learning. *17th International Conference on Advances in ICT for Emerging Regions, ICTer 2017 - Proceedings, 2018-Janua*, 263–270. <https://doi.org/10.1109/ICTER.2017.8257821>
- Meddeb, R., Jemili, F., Triki, B., & Korbaa, O. (2023). A deep learning-based intrusion detection approach for mobile Ad-hoc network. *Soft Computing*, 27(14), 9425–9439. <https://doi.org/10.1007/s00500-023-08324-4>
- Mohsin, A. H. (2022). Optimize Routing Protocol Overheads in MANETs: Challenges and Solutions: A Review Paper. In *Wireless Personal Communications* (Vol. 126, Issue 4). Springer US. <https://doi.org/10.1007/s11277-022-09843-3>
- Mukhametzyanov, F., Katasev, A. S., Akhmetvaleev, A. M., & Kataseva, D. V. (2019). The neural network model of DDoS attacks identification for information management fail. *International Journal of Supply Chain Management*, 8(5), 214–218.
- Naguib, M., Mohasseb, Y., Dahshan, H., & Elghandour, A. (2018). *Military Technical College Cairo , Egypt 11 th International Conference on Electrical Engineering ICEENG 2018 An Enhanced Transport Layer Protocol for Cognitive Mobile Ad Hoc Networks Proceedings*

- of the 11 th ICEENG Conference , 3-5 April , 2018. 3–5.
- Olusanya, M. O., & Vincent, O. R. (2020). A Manet-Based Emergency Communication System for Environmental Hazards Using Opportunistic Routing. *2020 International Conference in Mathematics, Computer Engineering and Computer Science, ICMCECS 2020*, 1–6. <https://doi.org/10.1109/ICMCECS47690.2020.240894>
- Parween, S., Islamia, J. M., Delhi, N., Hussain, S. Z., Islamia, J. M., & Delhi, N. (2023). *TCP Performance Enhancement in IoT and MANET: A Systematic Literature Review*. 10(4), 543–568. <https://doi.org/10.22247/ijcna/2023/223313>
- Perez-Diaz, J. A., Valdovinos, I. A., Choo, K. K. R., & Zhu, D. (2020). A Flexible SDN-Based Architecture for Identifying and Mitigating Low-Rate DDoS Attacks Using Machine Learning. *IEEE Access*, 8, 155859–155872. <https://doi.org/10.1109/ACCESS.2020.3019330>
- Poongothai, T., & Jayarajan, K. (2018). Intrusion Detection System for Mobile Ad Hoc Networks using Cross Layer and Machine Learning Approach. *International Journal of Computer Applications*, 179(34), 5–12. <https://doi.org/10.5120/ijca2018916680>
- Prasad, M., Tripathi, S., & Dahal, K. (2020). Intrusion Detection in Ad Hoc Network Using Machine Learning Technique. *Communications in Computer and Information Science*, 1317(December), 60–71. https://doi.org/10.1007/978-3-030-62625-9_6
- Prathyusha, D. J., & Kannayaram, G. (2021). A cognitive mechanism for mitigating DDoS attacks using the artificial immune system in a cloud environment. *Evolutionary Intelligence*, 14(2), 607–618. <https://doi.org/10.1007/s12065-019-00340-4>
- Rahman, M. A. (2025). Detection of Distributed Denial of Service Attacks based on Machine Learning Algorithms. *International Journal of Smart Home*, 14(2), 15–24. <https://doi.org/10.21742/ijsh.2020.14.2.02>
- Ramphull, D., Mungur, A., Armoogum, S., & Pudaruth, S. (2021). A review of mobile Ad hoc NETWORK (MANET) protocols and their applications. *Proceedings - 5th International Conference on Intelligent Computing and Control Systems, ICICCS 2021, Iccics*, 204–211. <https://doi.org/10.1109/ICICCS51141.2021.9432258>
- Safari, F., Savic, I., Kunze, H., Ernst, J., & Gillis, D. (2023). The Diverse Technology of MANETs: A Survey of Applications and Challenges. *International Journal of Future Computer and Communication*, 12(2), 37–48. <https://doi.org/10.18178/ijfcc.2023.12.2.601>

- Sharma, M., & Rashid, M. (2020). Security Attacks In MANET – A Comprehensive Study. *SSRN Electronic Journal*, April. <https://doi.org/10.2139/ssrn.3565860>
- Sivanesh, S., & Sarma Dhulipala, V. R. (2022). Analytical Termination of Malicious Nodes (ATOM): An Intrusion Detection System for Detecting Black Hole attack in Mobile Ad Hoc Networks. *Wireless Personal Communications*, 124(2), 1511–1524. <https://doi.org/10.1007/s11277-021-09418-8>
- Srilakshmi, U., Alghamdi, S. A., Vuyyuru, V. A., Veeraiah, N., & Alotaibi, Y. (2022). A Secure Optimization Routing Algorithm for Mobile Ad Hoc Networks. *IEEE Access*, 10, 14260–14269. <https://doi.org/10.1109/ACCESS.2022.3144679>
- Srivastava, S., Yadav, C. S., & Kumar, P. (2021). *Security Analysis Of Malicious Attacks In MANET Through Machine Learning Algorithm*. 18(6), 5866–5882.
- Su, T., Sun, H., Zhu, J., Wang, S., & Li, Y. (2020). BAT: Deep Learning Methods on Network Intrusion Detection Using NSL-KDD Dataset. *IEEE Access*, 8, 29575–29585. <https://doi.org/10.1109/ACCESS.2020.2972627>
- Sultan, M. T., El Sayed, H., & Khan, M. A. (2023). an Intrusion Detection Mechanism for Manets Based on Deep Learning Artificial Neural Networks (Anns). *International Journal of Computer Networks and Communications*, 15(1), 1–15. <https://doi.org/10.5121/ijcnc.2023.15101>
- Sworna, Z. T., Mousavi, Z., & Babar, M. A. (2023). NLP methods in host-based intrusion detection systems: A systematic review and future directions. *Journal of Network and Computer Applications*, 220. <https://doi.org/10.1016/j.jnca.2023.103761>
- Tang, L., Lu, Z., & Fan, B. (2020). Energy efficient and reliable routing algorithm for wireless sensors networks. *Applied Sciences (Switzerland)*, 10(5). <https://doi.org/10.3390/app10051885>
- Tchembe, M. X., & Tchoupe Tchendji, M. (2023). Building a publish/subscribe information dissemination platform for hybrid mobile ad-hoc social networks over android devices. *Revue Africaine de Recherche En Informatique et Mathématiques Appliquées*, Volume 36. <https://doi.org/10.46298/arima.9136>
- Thaljaoui, A. (2025). Intelligent network intrusion detection system using optimized deep CNN - LSTM with UNSW - NB15. *International Journal of Information Technology*. <https://doi.org/10.1007/s41870-025-02416-0>

- Tilwari, V., Maheswar, R., Jayarajan, P., Sundararajan, T. V. P., Hindia, M. N., Dimyati, K., Ojukwu, H., & Amiri, I. S. (2020). MCLMR: A Multicriteria Based Multipath Routing in the Mobile Ad Hoc Networks. *Wireless Personal Communications*, 112(4), 2461–2483. <https://doi.org/10.1007/s11277-020-07159-8>
- Tran, N., Chen, H., Bhuyan, J., & Ding, J. (2022). Data Curation and Quality Evaluation for Machine Learning-Based Cyber Intrusion Detection. *IEEE Access*, 10(Ml), 121900–121923. <https://doi.org/10.1109/ACCESS.2022.3211313>
- Tripathy, B. K., Jena, S. K., Reddy, V., Das, S., & Panda, S. K. (2021). A novel communication framework between MANET and WSN in IoT based smart environment. *International Journal of Information Technology (Singapore)*, 13(3), 921–931. <https://doi.org/10.1007/s41870-020-00520-x>
- Verma, P., Tapaswi, S., & Godfrey, W. W. (2020). An Adaptive Threshold-Based Attribute Selection to Classify Requests Under DDoS Attack in Cloud-Based Systems. *Arabian Journal for Science and Engineering*, 45(4), 2813–2834. <https://doi.org/10.1007/s13369-019-04178-x>
- Virupakshar, K. B., Asundi, M., Channal, K., Shettar, P., Patil, S., & Narayan, D. G. (2020). Distributed Denial of Service (DDoS) Attacks Detection System for OpenStack-based Private Cloud. *Procedia Computer Science*, 167(2019), 2297–2307. <https://doi.org/10.1016/j.procs.2020.03.282>
- Xue Chen, Giulia Tessari, Massimo Fabris, V. A. and M. F. (2021). Using Machine Learning for DoS Attacks Diagnostics. *Springer Nature Singapor*, August, 1–8. <https://doi.org/10.1007/978-3-030-68476-1>
- Yao, H., Fu, D., Zhang, P., Li, M., & Liu, Y. (2019). MSML: A novel multilevel semi-supervised machine learning framework for intrusion detection system. *IEEE Internet of Things Journal*, 6(2), 1949–1959. <https://doi.org/10.1109/JIOT.2018.2873125>
- Yi, S., Naldurg, P., & Kravets, R. (2001). Security-aware ad hoc routing for wireless networks. *In The Second ACM Symposium on Mobile Ad Hoc Networking & Computing, December*, 299. <https://doi.org/10.1145/501449.501464>
- Yusof, A. R. A., Udzir, N. I., Selamat, A., Hamdan, H., & Abdullah, M. T. (2017). Adaptive feature selection for denial of services (DoS) attack. *2017 IEEE Conference on Applications, Information and Network Security, AINS 2017, 2018-Janua(November)*, 81–

84. <https://doi.org/10.1109/AINS.2017.8270429>

Zoghi, Z., & Serpen, G. (2023). UNSW-NB15 computer security dataset: Analysis through visualization . *Security and Privacy*, May 2023, 1–29. <https://doi.org/10.1002/spy2.331>

APPENDIX A. SAMPLE CODE FOR DATASET TRAINING

This code was executed in jupyter notebook and exported as .py files

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing
from sklearn.preprocessing import LabelEncoder, Imputer,MinMaxScaler
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.decomposition import PCA
from sklearn.metrics import confusion_matrix,accuracy_score,precision_recall_fscore_support
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from keras.wrappers.scikit_learn import KerasClassifier
from keras.models import Sequential
from keras.layers import Dense
from keras.optimizers import Adam
from keras.utils.np_utils import to_categorical
from sklearn.ensemble import RandomForestRegressor
df=pd.read_csv('combined2.csv')

df_value=df[' Label'].value_counts()
df[' Label']=df[' Label'].apply({'DoS Hulk':'DoS', 'DoS GoldenEye':'DoS','DoS
Slowhttptest':'DoS','DoS
slowloris':'DoS' ,'BENIGN':'BENIGN' ,'DDoS':'DDoS', 'PortScan':'PortScan'}).get()
df2=df.drop_duplicates()
df2_value=df2[' Label'].value_counts()
datatype=df2.dtypes
df2['Flow Bytes/s']=df2['Flow Bytes/s'].astype('float64')
df2[' Flow Packets/s']=df2[' Flow Packets/s'].astype('float64')
```

```

NaN_values=df2.isnull().sum()
df2['Flow Bytes/s'].fillna(df2['Flow Bytes/s'].mean(),inplace=True)
print('The first read version of the dataset: \n',df_value)
print('First (row, Column) number of the dataset: {} '.format(df.shape))
print('Combining DoS dips in the Dataset Label and reducing noise:\n',df2_value)
print('Last (row, Column) number of the dataset: {} '.format(df2.shape))
dataset=pd.read_csv('dataset.csv')
dataset
DoS_df1=dataset[dataset[' Label']=='BENIGN']
DoS_df=DoS_df1.append(dataset[dataset[' Label']=='DoS'])
DoS_df
DDoS_df1=dataset[dataset[' Label']=='BENIGN']
DDoS_df=DDoS_df1.append(dataset[dataset[' Label']=='DDoS'])
DDoS_df
PortScan_df1=dataset[dataset[' Label']=='BENIGN']
PortScan_df=PortScan_df1.append(dataset[dataset[' Label']=='PortScan'])
PortScan_df
NA_df=dataset
NA_df[' Label']=NA_df[' Label'].apply({'DoS':'Anormal','BENIGN':'Normal',
,'DDoS':'Anormal', 'PortScan':'Anormal'}).get)
NA_df
def train_test_dataset(df):
    labelencoder = LabelEncoder()
    df.iloc[:, -1] = labelencoder.fit_transform(df.iloc[:, -1])
    X = df.drop([' Label'],axis=1)
    y = df.iloc[:, -1].values.reshape(-1,1)
    y=np.ravel(y)
    X_train, X_test, y_train, y_test = train_test_split(X,y, train_size = 0.7, test_size = 0.3,
random_state = 0, stratify = y)
    return X_train, X_test, y_train, y_test

```