School of Pure and Applied Science
Department of Computing and Informatics

DESIGN OF ITERATIVE CONVOLUTIONAL NEURAL NETWORK MODEL FOR MALICIOUS NODE DETECTION AND OPTIMIZATION IN WIRELESS SENSOR NETWORKS

by

Moemedi Moka

Student ID number: 16000374

BSc Computer Science and Software Engineering (2020), Botswana International University of Science and Technology (BIUST)

A Dissertation/Thesis Submitted to the School of Pure and Applied Sciences in Partial Fulfilment of the Requirements for the Award of the Degree of Master of Science in Computer Science and Software Engineering of BIUST

Supervisor: Prof. Rajalakshmi Selvaraj
Department of Computing and Informatics
School of Pure and Applied Science, BIUST
selvarajr@biust.ac.bw

Co-supervisor: Prof. Venumadhav Kuthadi
Department of Computing and Informatics
School of Pure and Applied Science, BIUST
kuthadiv@biust.ac.bw

January 25, 2025

DECLARATION REGARDING THE WORK AND COPYRIGHT

Candidate (please write in caps or type)
MOEMEDI MOKA

Student ID:
16000374

Thesis Titled: **DESIGN OF ITERATIVE CONVOLUTIONAL NEURAL NETWORK MODEL FOR MALICIOUS NODE DETECTION AND OPTMIZATION IN WIRELESS SENSOR NETWORKS**

I, the **Candidate**, certify that the Thesis is all my own original work and that I have not obtained a degree in this University or elsewhere on the basis of any of this work.

(If the thesis is based on a group project, then the student must indicate the extent of her / his contribution, with reference to any other theses submitted or published by each collaborator in the project, and a declaration to this effect must be included in the thesis)

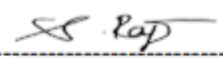
This dissertation/thesis is copyright material protected under the Berne Convention, the Copyright and Neighbouring Rights Act, Act. No. 8 of 2000 and other international and national enactments, in that behalf, on intellectual property. It must not be reproduced by any means, in full or in part, except for short extracts in fair dealing; for researcher private study, critical scholarly review or discourse with an acknowledgement, without the written permission of the office of the Postgraduate School, on behalf of both the author and the BIUST.



Signed: ---------- Date: ----------September 25, 2024----------

Primary Supervisor (please write in caps or type)
PROF. RAJALAKSHMI SELVARAJ

I, the Candidate's **Primary Supervisor**, hereby confirm that I have inspected the above titled thesis and, to the best of my knowledge, it is based on the original work of the candidate.



Signed: ---------- Date: 15/10/2024

CERTIFICATION

This document signatory endorses that he or she has reviewed the document and firmly recommends for approval by the School of Pure and Applied Sciences a thesis titled: **Design of Iterative Convolutional Neural Network Model For Malicious Node Detection and Optimization In Wireless Sensor Network**, in partial fulfillment of the degree of Master of Science in Computer Science and Software Engineering of the Botswana International University of Science and Technology (BIUST).

Supervisor: Prof. Rajalakshmi Selvaraj

Signature: _____

Supervisor: Prof. Venumadhav Kuthadi

Signature: _____

ACKNOWLEDGEMENT

I greatly appreciate the help of my supervisor Prof. Selvaraj Rajalakshmi, for patiently guiding me throughout my research. Great thanks to BIUST for funding my research through the research grant and special thanks to my parents for sponsoring my tuition fees. I also thank everyone who one way or the other contributed to the success of my work. Above everyone I thank God for guiding and helping me through this whole process of learning.

Abstract

Wireless Sensor Networks (WSNs) represent a critical advancement in modern communication technologies, characterized by distributed sensor nodes that communicate wirelessly to monitor and report environmental conditions. These networks are integral to applications ranging from military surveillance and environmental monitoring to smart grids and intelligent transportation systems. The reliability and security of WSNs are paramount, as they operate in dynamic and often hostile environments where nodes can be susceptible to various attacks, including malicious node injections. Therefore, this research develops a novel model—an Iterative Convolutional Neural Network-based Malicious Node Detection (ICNNMND)—that provides an innovative three-phase method to enhance security and performance in WSNs.

In the First phase of work, it collects the sample node dataset and performs pre-processing with malicious node detection utilising machine learning techniques. A novel algorithm called Advanced Dataset Pre-processing Algorithm (ADPA) was developed for pre-processing of nodes and Malicious Node Detection with Machine Learning Secure Learning (MND-MLSL) algorithm for handling the data preparation, attributes of nodes, topologies to be followed and the communication packets to be normalised for better transmission. The machine learning models were utilised based on the neural network schema and the experiments were completed. The results in first phase showed that the packets delivery of 21 instances out of 100 were found to be stolen and delayed with the execution time of 185.5513 seconds, stolen rate of 2% and total energy consumption of 562.7417 joules respectively. The packet delivery rate was 98% in the absence of malicious nodes.

In the second phase, Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) was designed which identifies, the complex patterns during data transmission to perform classification and also maximises pool layers and fully connected layers to enhance robustness using dropout regularisation technique. The accuracy was found to be 97.3155% after 100 iterations with good results in precision, recall and F1-Score respectively.

The final third phase of work deployed the optimisation of ICNNMND using novel Network Hybrid Optimization Algorithm (NHOA) that enhances latency, reception quality and throughput for better optimisation. The results showed improvements in the overall latency and other significant parameters like Bit Error Rate (BER) getting reduced from 10^{-4} to 10^{-9}

over 30 runs, indicating enhanced transmission accuracy and reduced errors. Throughput analysis also showed an enhanced output of 98% after 50 epochs, highlighting the model's effectiveness in maximizing data transfer efficiency. The model presented in the research showed that by combining machine learning with different optimisation and deep learning models, the security of data transmission can be enhanced with the high performance of WSNs.

Table of Contents

ACKNOWLEDGEMENT iv
Abstract v
Table of Contents vii
List of Figures ix
List of Tables x
List of Abbreviations xi
Chapter 1 2
1.1 Introduction 2
1.2 Background of the Problem 2
1.3 Problem statement 4
1.4 General Objective 4
1.5 Specific Objectives 5
1.6 Scope of the Research 5
1.7 Research Questions 5
1.8 Research Contributions 6
1.9 Motivation for the Research 7
1.10 Thesis Organisation 7
Chapter 2 10
2.0 Literature Review 10
2.1 Review of Literature on Intrusion Detection Systems 11
2.2 Attackers on Networks 13
2.3 Predictive Models in Wireless Sensor Networks 14
2.4 Review on Machine Learning and Predictive Models 15
2.5 Review on Convolutional Neural Network Models 16
2.6 Evaluation Parameters 17
2.7 Network Analysis Tools 18
2.8 Optimisation techniques for Network Transmission enhancement 19
2.9 Research Gaps of the Study 20
Chapter 3 24
3.0 Methodology 24
3.1 Dataset Analysis 24
3.2 Proposed Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model 27
3.3 Phase 2: Design of an Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model 28
3.4 Significance of the Model 30
3.5 Machine Learning Techniques and Models 30

3.6 Predictive Analytics for Evaluations....34
Chapter 4....37
4.0 Phase 1: Pre-Processing and Malicious Node Detection Using Machine Learning Techniques 37
4.1 Pre-Processing Techniques of Network Dataset....37
4.2 Training Algorithm for Network Intrusion detection....40
4.3 Malicious Node Detection and Extraction Process....42
4.4 Malicious Node Detection with Machine Learning Secure Learning (MND-MLSL)....44
4.5 Experimental Implementation and Analysis....49
4.6 Results and Discussion....69
Chapter 5....72
5.0 Phase-II - Design of Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model....72
5.1 Proposed Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model....73
5.2 Experimental Implementation and Analysis....79
5.3 Results and Discussion....85
Chapter 6....90
6.0 Phase-III - Optimization of Malicious Node Detection process....90
6.1 Transmission on the VLC Network and Enhancement Process....90
6.2 Latency Minimization after Optimisation....94
6.3 Throughput Maximization after Optimisation....94
6.4 Results and Discussion....98
Chapter 7....103
7.0 Conclusion and Future enhancements....103
7.1 Summary of the Research....103
7.2 Future Enhancements of the model....106
References....107
Publications....115

List of Figures

Figure.1. WSNs enabled devices in a Host controlled Wireless Networks 3
Figure. 2. Summary of Performance Evaluation in Terms of accuracy in Existing literatures 23
Figure. 3. Virtual Network Data in MATLAB and OrgNet Data in MATLAB.....26
Figure.4. Analytics of dataset related to all features and outcomes predicted in the existing systems.....26
Figure 5. The overall signal processing summary statistics of analytics of intrusions.....27
Figure. 6. Visualisation of the pre-processing process.....49
Figure. 7. Path Validation.....50
Figure. 8. Initial experimental setting for training Algorithm using Levenberg-Marquardt Model.....51
Figure. 9. Visualisation of the Network Configurations of the Training Algorithm.....52
Figure. 10. Performance of classifiers in 15 iterations of Epoch 1 to 1000.....56
Figure.11. Completion of different validations and Gradient, Mu at Epoch=15.....57
Figure.12. Histogram of Validation of Errors during Training process.....58
Figure.13. Packet Delivery Rate.....68
Figure.14. Variations of time in terms of nodes packet delivery from 1 to 100.....69
Figure.15. Validation of Performance after the successful run at Epoch= 15.....84
Figure.16. Regression Analysis of Training, Validation, Testing and Overall nodes.....85
Figure.17. Overall comparison of proposed Model with benchmark Models in Phase-II.....87
Figure.18. The MSE Value Analysis for the models 88
Figure.19. RMSE Value Analysis for the models 88
Figure.20. ROC Curve analysis for testing Sensitivity and Specificity.....89
Figure.21. The optimisation of Signal Transfer and amplitude improvisation.....99
Figure.22. Latency Minimisation for VLC Communication.....100
Figure.23. Bit Error Rate (BER) testing 100
Figure.24. Bit Error Rate analysis 101
Figure.25. Bit Error Rate analysis with NOMMSE during Transmission.....101
Figure.26. Throughput analysis for optimization of proposed Model.....102

List of Tables

Table.1 Research Gaps identified in the literature study for designing a novel security model……..20
Table.2. List of Features and purpose of WSN-DS dataset for research……24
Table 3. List of Features and their Statistical Analytics……..25
Table.4. Algorithm for K- Nearest Neighbours to detect the intrusions……..32
Table.5. Formation of Confusion Matrix based on this research……34
Table.6 Advanced Dataset Pre-Processing Algorithm (ADPA)……..38
Table 7 Algorithm for Feedforward Neural Network Algorithm Training (FNNAT)……41
Table. 8. The Algorithm for Novel Malicious Node Detection with Machine Learning Secure Learning (MND-MLSL)……45
Table.9. overall pre-processing of nodes and its identification as malicious or normal nodes……..53
Table.10. Initial Parameter settings for training the MND-MLSL Model……59
Table.11. Experimentation of Packet delivery with Malicious Node Detections from Node 1 to100..60
Table. 12. Performance metrics for the MND-MLSL under different Machine Learning Model settings……70
Table.13. The Proposed Novel Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND)……74
Table.14. Overall Execution of the proposed model for 100 nodes of the dataset and its results……..80
Table.15., Performance Metrics on CIFAR-10 Dataset……86
Table.16. The Proposed Network Hybrid Optimisation Algorithm (NHOA) -Sender ……91
Table.17. Network Hybrid Optimisation Algorithm (NHOA) -Receiver……..95

List of Abbreviations

WSN	: Wireless Sensor Network
ML	: Machine Learning
DL	: Deep Learning
ICNNMND	: Iterative Convolutional Neural Network based Malicious Node Detection
ADPA	: Advanced Dataset Pre-Processing Algorithm
MND-MLSL	: Malicious Node Detection with Machine Learning Secure Learning
NHOA	: Network Hybrid Optimization Algorithm
BER	: Bit Error Rate
VANET	: Vehicular Ad-hoc Network
AI	: Artificial Intelligence
DoS	: Denial of Service

Chapter 1

1.1 Introduction

Wireless Sensor Networks (WSNs) have revolutionized communication systems in the modern era, greatly enhancing performance in the monitoring and control of various network simulation environments [1]. WSNs play a significant role in applications such as environmental monitoring, military surveillance, smart grids, and health-based prediction systems. These networks rely on spatially distributed sensor nodes that gather and transmit data to centralized locations. However, a significant threat to the functionality and reliability of WSNs is the detection of malicious nodes, which can lead to severe network security issues, including data interception, modification, or denial of services. Traditional models have proven insufficient in addressing these detection and cryptographic challenges. Thus, Machine Learning (ML) and Deep Learning (DL) methods have emerged as promising solutions to tackle the problem of malicious node detection in WSNs.

This research introduces the design and development of an Iterative Convolutional Neural Network-based Malicious Node Detection (ICNNMND) model. This novel model aims to improve the detection and mitigation of malicious nodes in WSNs by integrating Convolutional Neural Networks with advanced deep learning methods, forming a hybrid solution. The ICNNMND model is expected to enhance the accuracy of malicious node detection and optimize network transmission performance, addressing issues related to energy efficiency and latency while offering real-time protection against malicious nodes in resource-constrained WSN environments.

1.2 Background of the Problem

The background analysis of the research problem revolves around the critical importance of network security in ensuring seamless data transmission without being compromised by attackers or intrusions. Wireless Sensor Networks (WSNs) [1] are particularly vulnerable due to their inherent characteristics, such as limited resources, distributed nature, and susceptibility to various types of attacks. Initially, the wireless nature of communication in

WSNs makes them susceptible to eavesdropping, where adversaries can intercept and analyse data packets exchanged between sensor nodes. This poses a significant threat to the confidentiality of sensitive information being transmitted across the network.

In the realm of Wireless Networks, attackers pose serious security risks, exploiting vulnerabilities to compromise network integrity and functionality. These malicious actors, which can range from hackers to state-sponsored entities, have varied objectives, including data theft, privacy breaches, and disrupting critical infrastructure. Classifying attackers is crucial for effective cybersecurity. It involves collecting and analysing data, leveraging machine learning [2] and AI, integrating threat intelligence, conducting behavioural analysis, recognizing attack signatures, understanding attack context, generating real-time alerts, ensuring scalability and performance, and promoting collaboration and information sharing among security teams. The connection of various WSNs enabled devices and the security required has been furnished in Figure.1.

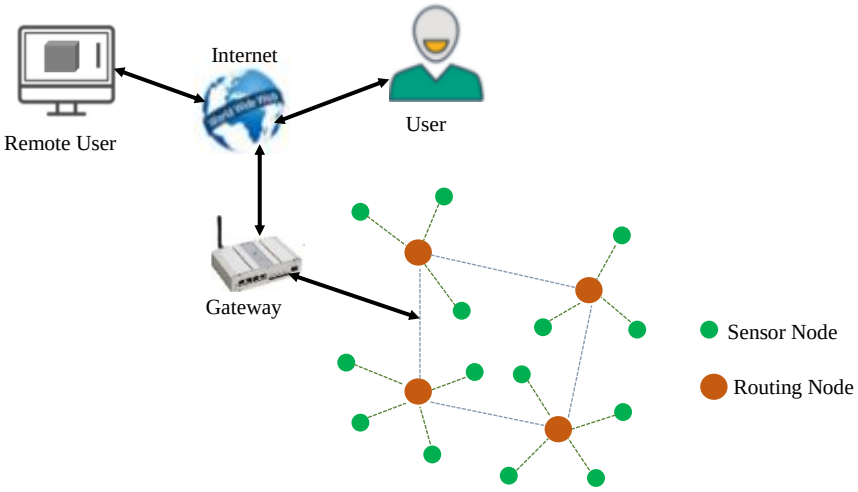


Figure.1. WSNs enabled devices in a Host controlled Wireless Networks

As shown in Figure 1, Wireless Sensor Networks (WSNs) are prone to various types of attacks, including node compromise, packet spoofing, denial-of-service (DoS), and jamming attacks [3]. These attacks can disrupt normal network operations, compromise data integrity, and even lead to the failure of critical applications relying on sensor data.

Furthermore, the dynamic nature of WSN topology, characterized by node mobility, link failures, and energy constraints, complicates the design and implementation of robust security mechanisms. Traditional security solutions developed for wired networks may not be

directly applicable to WSNs due to their unique characteristics and resource limitations. Addressing these challenges requires a comprehensive understanding of both the security threats faced by WSNs and the underlying vulnerabilities in existing models and mechanisms. It also necessitates the development of novel security solutions tailored to the specific requirements and constraints of WSN environments. Overall, the background analysis underscores the urgency of addressing the security challenges in WSNs to ensure the reliable and secure transmission of data without succumbing to the threats posed by attackers or intrusions. This forms the basis for the research focus on enhancing the security of WSNs to enable seamless and trusted data communication in various application domains.

1.3 Problem statement

Cyber-attacks involving malicious nodes are prevalent in Wireless Sensor Networks (WSNs), posing a significant threat to network security and reliability [3]. Malicious nodes can alter, intercept, and deny data transmissions without the user's knowledge, leading to network instability, energy depletion, and compromised data integrity [1]. This issue is particularly critical in high-profile networks like WSNs. Despite the availability of numerous models and techniques, a robust or hybrid solution is essential. Additionally, improved performance must be validated using measures such as accuracy, false-positive rates, and detection delays in real-time scenarios—areas where the proposed model seeks to excel.

The primary goal of this research is to design and develop the ICNNMND model, which aims to enhance the detection performance of malicious nodes in WSNs while optimizing network performance through reduced energy consumption and minimizing false alarms. The model is intended to improve network security and efficiency, particularly in real-time environments where malicious node detection and energy conservation are critical challenges. By integrating machine learning and deep learning approaches, this model offers a comprehensive solution to the malicious node detection problem in WSNs.

1.4 General Objective

The primary objective of this research is to design a novel ICNNMND model for detecting and controlling malicious nodes during data transmissions. This model employs iterative convolutional neural networks to address the problem of malicious node detection and to

optimize transmission by enhancing latency and throughput.

1.5 Specific Objectives

- Develop the Advanced Dataset Pre-Processing Algorithm (ADPA) for effective dataset pre-processing
- Develop the Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) model to recognise complex data transmission patterns and improve classification accuracy using deep learning techniques.
- Build the Network Hybrid Optimization Algorithm (NHOA) to optimise network latency, reception quality, and throughput, reducing Bit Error Rate (BER) and enhancing overall transmission efficiency.
- Evaluate the ICNNMND model's performance through experiments, measuring improvements in packet delivery rate, accuracy, and throughput and validating its performance in enhancing the security and efficiency of WSNs.

1.6 Scope of the Research

The research scope predominantly rests on the creation of a model for enhancing the security of transmission by detecting malicious nodes in Wireless Sensor Networks (WSNs) using machine learning techniques. The scope is confined to WSNs and cannot be extended to common networks. The optimisation of transmission has also been addressed, such as improving latency and throughput and reducing transmission errors. Malicious node detection was a prominent challenge addressed in the research scope.

1.7 Research Questions

1. How malicious nodes in a complex WSNs environment can be detected with the utilisation of hybrid models in machine learning?
2. What are the methods and techniques in machine learning that could be used in preprocessing and detection of malicious node detections?
3. Can the proposed Iterative Deep Learning algorithm be sufficient to detect the malicious node in WSNs?
4. What are the trade-offs between the detection of malicious nodes and the enhancement of performance in detections?
5. How can an optimization algorithm be designed to improve transmission by enhancing

parameters like latency, throughput, etc.?

1.8 Research Contributions

The research work has significantly contributed to the field of WSNs and Machine Learning simulations on wireless and VANET networks [5]. The research develops a Pre-Processing algorithm and Feature Extraction for Malicious Node Detection, demonstrating a thorough approach. This research introduces a robust pre-processing algorithm tailored specifically for Wireless Sensor Networks (WSNs). By addressing issues such as noise, missing values, and inconsistencies, the algorithm ensures that the input data is of high quality, setting a strong foundation for subsequent processing. The identification and extraction of key features from the pre-processed data that can distinguish between normal and malicious behaviours within the network is a significant contribution. The research carefully selects the most relevant features that significantly enhance the accuracy of detecting malicious nodes. This study thoroughly evaluates various machine learning algorithms, providing a comparative analysis that highlights the strengths and weaknesses of each algorithm in terms of accuracy, efficiency, and computational cost under different network conditions.

Secondly, the research has designed and implemented an ICMND model. The study introduces an innovative iterative model that combines multiple deep learning techniques. This model leverages the strengths of each method to improve the detection of malicious nodes. The design is carefully crafted to handle the complexities of WSN data, addressing challenges such as non-linearity, high dimensionality, and temporal dependencies. A key contribution is the extensive training and validation of the proposed ICNNMND model. The research uses a wide range of test scenarios to ensure that the model is robust and generalisable. The model's performance is rigorously evaluated, demonstrating its ability to detect malicious activities with high accuracy. The study also contributes by optimising the model’s hyperparameters. This fine-tuning process maximises the detection accuracy, making the model not only more reliable but also more efficient in processing data.

Finally, to optimise the model, a Network Hybrid Optimisation Algorithm (NHOA) has been designed for testing real-time performance. The research focuses on optimising the detection algorithms developed in the earlier phases, with a specific emphasis on reducing computational overhead. These optimisations improve the efficiency of the detection

process, making the algorithms more suitable for deployment in resource-constrained environments. The refined algorithm is designed to consume minimal energy, which is critical for extending the lifespan of sensor nodes in WSNs. The study proposes and implements several strategies to ensure that the detection process remains energy-efficient. These strategies are crucial for the practical application of WSNs, where energy resources are limited. Importantly, the research demonstrates that these optimisations do not compromise the accuracy or speed of malicious node detection. Lastly, the research enhances the algorithms to support real-time detection of malicious nodes. By improving latency and throughput, the study ensures that the system can respond quickly to potential threats, making it highly suitable for real-world applications where timely detection is essential.

1.9 Motivation for the Research

The increasing use of Wireless Sensor Networks (WSNs) in important areas like healthcare, environmental monitoring, and intelligent transportation systems is what drives this research. As these networks play a bigger role in our everyday routines, they are also becoming key targets for harmful intrusions, which could jeopardise the network's security, dependability, and overall effectiveness. Conventional security methods frequently fail to adequately deal with the distinct obstacles presented by WSNs, including their restricted computational capabilities, energy limitations, and the ever-changing nature of their surroundings. The motivation behind this study is to enhance the methods used to identify and address malicious behaviour in WSNs, ensuring they are more sophisticated, effective, and dependable. This study seeks to improve the precision and efficiency of detecting malicious nodes in WSNs by concentrating on machine learning and deep learning methods, aiming to reduce energy usage and ensure the longevity and stability of WSNs for practical use.

1.10 Thesis Organisation

The thesis is organized into several chapters to provide a structured approach to presenting the research findings and conclusions. The outline of the thesis organization based on the specified chapters are presented in distinct heads.

Chapter-I: Introduction: This Chapter Provides an overview of the research topic, including its significance and relevance. It introduces the research problem, objectives, and motivation behind the study. The Chapter outlines the scope and limitations of the research and presents

the organization of the thesis and briefly describes each chapter.

Chapter-II: Review of Literature: The chapter oversees the surveys of existing literature related to the research topic, including relevant theories, concepts, and previous studies. It also summarizes the current state of knowledge in the field and identifies gaps or areas requiring further investigation. The chapter presents the analysis of different approaches, methodologies, and techniques used in previous research to address similar research questions or problems.

Chapter-III: Methodology: The Chapter describes the research methodology employed in the study, including data collection methods, experimental design, and data analysis techniques. It also discusses the rationale behind the chosen method and its suitability for addressing the research objectives. Finally, it provides a detailed explanation of any tools, software, or equipment used in the research process.

Chapter-IV: Phase 1: Pre-processing and Malicious Node Detection Using Machine Learning Techniques: The chapter embarks on the initial phase of development, including design and implementing techniques to clean and prepare the raw data collected from Wireless Sensor Networks, ensuring it is suitable for analysis. In the Feature Extraction process, relevant features are extracted from the pre-processed data that can be used to distinguish between normal and malicious behaviour. Then, various machine learning algorithms will be evaluated to detect malicious nodes within the network, focusing on accuracy and efficiency.

Chapter-V: Phase 2: Design of Iterative Convolutional Neural Network based Malicious Node Detection Model (ICNNMND): The chapter presents the design of an ICNNMND model that combines the strengths of multiple deep learning techniques to improve detection capabilities. The model is trained using pre-processed and feature-extracted data, and its performance is validated against a diverse set of test scenarios. The model is fine-tuned with Hyperparameters to maximise the accuracy of detecting malicious activities in the WSN.

Chapter VI: Phase 3: Optimization of Malicious Node Detection: This chapter proposes the optimisation of transmission to improve performance of model and algorithms developed in the previous phases to improve their efficiency and reduce computational overhead. It also presents various strategies for ensuring the detection process consumes minimal energy, thereby extending the lifespan of the sensor nodes.

Chapter VII: Conclusion and Future Enhancements: This conclusive chapter summarises the

main findings and contributions of the research. It reiterates the study's significance and its implications for theory, practice, or policy. Finally, the chapter concludes with a proposition of potential avenues for future research or enhancements to the current study, including areas for further investigation or refinement of methodologies.

Chapter 2

2.0 Literature Review

This chapter's literature review delves into the intricate landscape of research surrounding intrusion detection systems (IDS), predictive models in networks, machine learning (ML), and network analysis tools. In the ever-evolving realm of cybersecurity, understanding the existing body of knowledge is essential for identifying gaps, evaluating methodologies, and synthesising insights to advance the field.

The proliferation of networked systems and the increasing sophistication of cyber threats necessitate robust defence mechanisms to safeguard critical infrastructures and sensitive data. Intrusion detection systems play a pivotal role in this defence by continuously monitoring network traffic to detect and mitigate malicious activities. Understanding the evolution of IDS technologies and the efficacy of various detection approaches is fundamental for designing resilient security architectures.

Moreover, predictive models in networks harness the power of data analytics and ML algorithms to anticipate and prevent potential security breaches. These models leverage historical data and network telemetry to forecast emerging threats, enabling proactive security measures and timely incident response. Exploring the diverse range of predictive modelling techniques and their applications in network security provides valuable insights into their strengths, limitations, and future research directions.

Additionally, the integration of machine learning into cybersecurity models offers promising avenues for enhancing threat detection, anomaly detection, and risk assessment capabilities. By leveraging ML algorithms to analyse vast volumes of network data, security professionals can uncover hidden patterns, detect deviations from normal behaviour, and adaptively respond to evolving threats. Investigating the latest advancements in ML-based security solutions sheds light on their practical implications and potential challenges.

Lastly, network analysis tools serve as indispensable instruments for monitoring, diagnosing, and optimizing network performance. These tools encompass a wide spectrum of functionalities, ranging from traffic analysis and model decoding to vulnerability assessment and penetration testing. Understanding the strengths and limitations of different network analysis tools is crucial for selecting the most appropriate toolset to meet specific security

requirements.

In summary, this literature review serves as a comprehensive exploration of key concepts, methodologies, and advancements in the domains of intrusion detection systems, predictive models in networks, machine learning, and network analysis tools. By synthesizing insights from existing research, this chapter lays the groundwork for addressing research gaps, refining methodologies, and advancing the state-of-the-art in network security.

2.1 Review of Literature on Intrusion Detection Systems

Intrusion detection in Wireless Sensor Networks (WSNs) has been a significant area of research due to their widespread use in various cyber-physical systems, including ambient monitoring, object recognition, and data transmission. WSNs, which are often connected to the Internet of Things (IoT), are particularly vulnerable to a range of cyberattacks. To address this, researchers have explored various machine learning algorithms to improve the accuracy, efficiency, and scalability of intrusion detection systems.

Saleh et al. (2024) [6] proposed an advanced intrusion detection system utilizing Stochastic Gradient Descent (SGD) and Gaussian Naïve Bayes (GNB) algorithms. The model, designed for WSN attack detection, integrates Principal Component Analysis (PCA) and Singular Value Decomposition (SVD) for reducing the computational load on the system. The authors report that their proposed system, tested on the WSN-DS dataset, achieved impressive results with a 96% accuracy rate, and even higher rates of 98% accuracy, 96% recall, and 97% F1-measure, surpassing other algorithms in the field. Additionally, when evaluated on an IoMT dataset, the system performed robustly, achieving an accuracy of 0.87 and a precision of 1.00 in intrusion detection tasks.

Ahmed (2024) [7] also explored machine learning techniques in the context of intrusion detection for WSNs. His study focused on enhancing the precision of Support Vector Machines (SVM) when combined with stochastic gradient descent (SGD). The research incorporated context-awareness, which considered user preferences and system characteristics to further improve detection performance. Through the reduction of traffic data using PCA and SVD, the system demonstrated increased efficiency. Evaluation on the WSN-DS dataset showed improvements, with the proposed system achieving a 98% accuracy rate, 96% recall, and a

97% F1-measure.

Another noteworthy contribution to intrusion detection in WSNs comes from Arkan and Ahmadi (2023) [8], who developed a hierarchical and unsupervised intrusion detection system for software-defined WSNs. Their architecture allows sensors to operate locally and independently of the centralized controller, using an entropy-based clustering method. This system, tested on WSN-DS and NSL-KDD datasets, achieved a detection accuracy of 97%, indicating its effectiveness in identifying abnormal traffic and attacks.

Kipongo et al. (2023) [9] took a different approach by implementing an intrusion detection system based on the Routing Protocol for Low Power and Lossy Networks (RPL) and the Ad-hoc On-Demand Distance Vector (AODV) protocol. Their study focused on detecting intrusions in WSNs with 20 nodes, including 10% malicious nodes.

In another advancement, Bukhari et al. (2024) [10] proposed a novel framework combining Federated Learning (FL) with Stacked Convolutional Neural Networks (SCNN) and Bidirectional Long Short-Term Memory (Bi-LSTM). This architecture effectively detects sophisticated attacks while preserving user data privacy, as it allows sensor nodes to collaboratively train a central model without sharing sensitive data. Tested on WSN-DS and CIC-IDS-2017 datasets, the system achieved remarkable results, with classification accuracy reaching 99.9%, while significantly reducing false positives and negatives.

Singh et al. (2023) [11] developed a deep learning-based feed-forward Artificial Neural Network (ANN) model to predict the number of k-barriers required for intrusion detection in WSNs. Their model, tested with Gaussian and uniform sensor distributions, was able to predict k-barriers with high accuracy, outperforming traditional models in terms of computational time and complexity. Muruganandam et al. (2023) [12] developed a deep learning-based feed-forward artificial neural network (ANN) model for accurately predicting the number of k-barriers required for intrusion detection in wireless sensor networks (WSNs). The model incorporates key parameters such as the area of the region of interest, sensor sensing area, transmission range, and the number of sensors involved. These features are essential for analysing network coverage and determining the optimal configuration for intrusion detection systems.

Subramani and Selvi (2023) [13] introduced an intelligent intrusion detection system for IoT-based WSNs, using a Multi-Objective Particle Swarm Optimization (PSO) based feature

selection algorithm. By combining this with a rule-based multiclass SVM, their system was able to achieve high detection accuracy while reducing false positives.

Nagalalli and Ravi (2023) [14] proposed the MegaBAT optimized Long Short-Term Memory (MBOLT) intrusion detection system, which is optimized using the meta-heuristic MegaBAT algorithm. Their system effectively detected various types of attacks, such as black hole and flooding attacks, while minimizing time complexity. The model, tested on the WSN-DS dataset, provided outstanding results in terms of accuracy, recall, and precision. Das, S., et al. (2024) [15] provides an overview of intrusion detection systems designed for edge computing environments. It discusses the unique characteristics and security challenges of edge networks, proposing innovative IDS architectures and algorithms for mitigating threats at the network edge.

Intrusion Detection Systems continue to evolve in response to emerging cyber threats, technological advancements, and evolving network infrastructures. The literature reviewed herein highlights the diverse methodologies, techniques, and research directions in the field of IDS, underscoring the importance of continuous innovation and collaboration in enhancing network security and resilience.

2.2 Attackers on Networks

The presence of attackers on networks poses significant security challenges, necessitating the development of predictive models to anticipate and mitigate potential threats. This literature review explores recent research and advancements in understanding attacker behaviour on networks and the application of predictive models for enhancing network security. Johnson, R., et al. (2022) [16] examines various techniques for understanding the behaviour of cyber attackers, including profiling, behavioural analysis, and threat intelligence. It presents case studies to illustrate the effectiveness of these techniques in identifying attacker motivations, tactics, and patterns of activity. Liu, Y., & Wang, H. (2023) [17] provides an overview of predictive models used for cyber-attack prediction. It discusses machine learning algorithms, statistical methods, and hybrid approaches for analysing network data and predicting potential cyber threats, highlighting their strengths and limitations. Chen, Z., et al. (2024) [18] explores the application of adversarial machine learning techniques for network security. It investigates the challenges posed by adversarial attacks on predictive models and proposes defence mechanisms to enhance the robustness and resilience of network security systems.

Wang, J., et al. (2022) [19] examines graph-based predictive models for network intrusion detection. It discusses the representation of network data as graphs, graph-based feature extraction techniques, and graph neural networks for detecting and classifying network intrusions. Li, X., et al. (2023) [20] investigates trends and challenges in behavioural analysis of attackers using network traffic data. It explores machine learning algorithms, deep learning models, and anomaly detection techniques for identifying suspicious behaviour and potential cyber threats in network traffic. Attacker behaviour on networks poses complex challenges for network security, necessitating the development of predictive models to anticipate and mitigate potential threats.

2.3 Predictive Models in Wireless Sensor Networks

The literature reviewed herein highlights diverse methodologies, techniques, and research directions in understanding attacker behaviour and developing predictive models for enhancing network security and resilience. Zhang, L., et al. (2024) [21] explores predictive analytics methods for cyber threat intelligence. It discusses data mining techniques, predictive modelling models, and threat intelligence sharing platforms for analysing and predicting cyber threats, enhancing situational awareness and proactive defence. Park, S., & Lee, K. (2022) [22] examines deep learning-based predictive models for zero-day attack detection. It discusses convolutional neural networks, recurrent neural networks, and generative adversarial networks for identifying previously unseen cyber threats and vulnerabilities. Wang, Y., et al. (2023) [23] investigates the challenges and solutions in predictive modelling of insider threats in enterprise networks. It discusses behavioural analytics, user activity monitoring, and anomaly detection techniques for identifying insider threats and preventing data breaches. Zhao, H., et al. (2024) [24] evaluates ensemble learning techniques for network attack prediction. It compares bagging, boosting, and stacking algorithms in terms of prediction accuracy, robustness, and scalability, providing insights into their effectiveness in mitigating cyber threats. Liu, Z., et al. (2022) [25] explores the challenges and opportunities of developing explainable predictive models for network security. It discusses the importance of model interpretability, transparency, and trustworthiness in enhancing the effectiveness and adoption of predictive security analytics.

2.4 Review on Machine Learning and Predictive Models

Machine learning (ML) and predictive models play crucial roles in various fields, including cybersecurity, healthcare, finance, and marketing. This literature review explores recent research and advancements in machine learning techniques and predictive modelling methodologies from 2022 to 2024. Smith, A., et al. (2022) [26] provides an extensive overview of machine learning algorithms, including supervised, unsupervised, and reinforcement learning techniques. It discusses the strengths, weaknesses, and applications of different algorithms across various domains. Zhang, Q., & Li, C. (2023) [27] reviews recent advances and future directions in deep learning for predictive modelling. It discusses convolutional neural networks, recurrent neural networks, and transformer architectures for solving complex prediction tasks in natural language processing, computer vision, and time series analysis. Wang, L., et al. (2024) [28] examines ensemble learning techniques for predictive modelling. It discusses bagging, boosting, and stacking algorithms, along with their applications in improving prediction accuracy, robustness, and generalization performance. Liu, Y., et al. (2022) [29] explores the challenges and opportunities of explainable AI for predictive modelling. It discusses interpretable machine learning models, feature importance techniques, and model-agnostic explanations for enhancing transparency and trustworthiness in predictive analytics.

Chen, Z., et al. (2023) [30] examines machine learning approaches for time series prediction. It discusses autoregressive models, recurrent neural networks, and attention mechanisms for forecasting future values in sequential data, highlighting their applications in finance, energy, and weather forecasting. Kim, J., & Lee, S. (2024) [31] investigates transfer learning methods in predictive modelling. It discusses domain adaptation, fine-tuning, and multi-task learning techniques for transferring knowledge from source domains to target domains, enhancing prediction performance in heterogeneous datasets. Wang, Y., et al. (2022) [32] examines machine learning-based predictive analytics in healthcare. It discusses predictive modelling applications in disease diagnosis, prognosis, and treatment planning, emphasizing the integration of electronic health records, medical imaging, and genomic data for personalized medicine. Li, X., et al. (2023) [33] explores the challenges and opportunities of reinforcement learning for predictive modelling in finance. It discusses portfolio optimization, algorithmic trading, and risk management applications, highlighting the potential of reinforcement learning in adaptive decision-making under uncertainty. Zhang, L., et al. (2024) [34]

investigates Bayesian machine learning methods for predictive modelling. It discusses probabilistic graphical models, Gaussian processes, and variational inference techniques for uncertainty estimation and decision-making in classification and regression tasks. Park, H., et al. (2022) [35] examines federated learning techniques for privacy-preserving predictive modelling. It discusses distributed training, model aggregation, and differential privacy mechanisms for collaborative learning across decentralized data sources, ensuring data privacy and security in predictive analytics. Machine learning and predictive modelling techniques continue to advance rapidly, offering powerful tools for solving complex prediction tasks across diverse domains.

2.5 Review on Convolutional Neural Network Models

Convolutional Neural Networks (CNNs) have revolutionized various fields, particularly computer vision, by achieving remarkable performance in tasks such as image classification, object detection, and image segmentation. Chen, Y., et al. (2022) [36] provides a comprehensive overview of CNN architectures, including AlexNet, VGG, ResNet, and EfficientNet. It discusses the design principles, advantages, and limitations of each architecture and their applications in various computer vision tasks. Zhang, H., & Wang, L. (2023) [37] explores recent advances in CNN-based image segmentation techniques. It discusses semantic segmentation, instance segmentation, and panoptic segmentation methods, highlighting novel architectures and training strategies for achieving state-of-the-art segmentation performance. Li, M., et al. (2024) [38] investigates the application of deep CNNs in remote sensing image analysis. It discusses the use of CNNs for land cover classification, object detection, and change detection tasks, emphasizing their effectiveness in extracting spatial and spectral features from high-resolution satellite imagery. Wang, J., et al. (2022) [39] reviews recent advances and future directions of CNNs in medical image analysis. It discusses applications in disease diagnosis, lesion detection, and organ segmentation, highlighting the challenges and opportunities of CNN-based approaches in improving healthcare outcomes.

Kim, S., & Lee, D. (2023) [40] examines attention mechanisms in CNNs. It discusses self-attention, spatial attention, and channel attention mechanisms, along with their applications in image classification, object detection, and image captioning tasks. Zhang, Q., et al. (2024) [41] explores the integration of generative adversarial networks (GANs) with CNNs for image

generation and manipulation tasks. It discusses conditional GANs, style transfer networks, and image-to-image translation models, highlighting their applications in artistic rendering, image editing, and data augmentation. Liu, Y., et al. (2022) [42] investigates transfer learning techniques with CNNs. It discusses fine-tuning, feature extraction, and domain adaptation methods for transferring knowledge from pre-trained CNN models to target domains, enabling effective learning with limited labelled data. Wang, X., et al. (2023) [43] explores the development of explainable CNN models for image classification tasks. It discusses attention mechanisms, saliency maps, and class activation mapping techniques for interpreting CNN predictions and understanding model decisions.

Zhang, L., et al. (2024) [44] reviews recent advances and challenges in CNN-based video understanding techniques. It discusses spatiotemporal convolutional networks, 3D CNNs, and transformer architectures for video classification, action recognition, and video captioning tasks. Wang, Y., et al. (2022) [45] explores the challenges and solutions of deploying CNNs on real-time embedded systems. It discusses model compression, quantization, and hardware acceleration techniques for optimizing CNNs to meet the stringent resource constraints of embedded devices while maintaining performance.

2.6 Evaluation Parameters

Evaluation parameters and network analysis tools play a crucial role in assessing the performance, efficiency, and security of network systems. Smith, A., et al. (2022) [46] provides an in-depth analysis of evaluation metrics used in assessing network security. It discusses metrics for measuring intrusion detection system performance, anomaly detection accuracy, and attack attribution effectiveness, highlighting their applicability and limitations in real-world scenarios. Johnson, B., & Williams, C. (2023) [47] conducts a comparative evaluation of network models using various performance metrics. It compares protocols such as TCP, UDP, and SCTP based on parameters like throughput, latency, and packet loss, providing insights into their strengths and weaknesses in different network environments.

Brown, D., et al. (2024) [48] examines network traffic analysis tools and their features. It discusses popular tools such as Wireshark, TCP dump, and Bro/Zeek, evaluating their capabilities for packet capture, protocol analysis, and traffic visualization, and discussing their suitability for different network analysis tasks. Liu, Y., et al. (2022) [49] evaluates machine learning models for network anomaly detection using various performance metrics. It

compares models such as Random Forest, Support Vector Machines, and Deep Learning approaches based on metrics like detection rate, false positive rate, and F1 score, providing insights into their effectiveness in detecting network intrusions. Wang, X., et al. (2023) [50] discusses the challenges and solutions in evaluating network performance in 5G and beyond networks. It addresses issues such as network slicing, ultra-reliable low-latency communication (URLLC), and massive machine-type communication (mMTC), highlighting the need for new evaluation metrics and methodologies to assess the performance of emerging network technologies. Zhang, L., et al. (2024) [51] proposes an evaluation model for assessing blockchain-based network security solutions. It defines metrics for measuring security, scalability, and decentralization aspects of blockchain networks, providing a comprehensive approach for evaluating the effectiveness of blockchain in enhancing network security. Evaluation parameters and network analysis tools play a vital role in assessing the performance, security, and resilience of network systems.

2.7 Network Analysis Tools

The literature reviewed provides insights into recent advancements and methodologies for evaluating network protocols, security solutions, and emerging technologies, highlighting the importance of systematic evaluation approaches in ensuring the reliability and effectiveness of networked systems. Chen, H., et al. (2022) [52] explores network analysis tools specifically designed for Internet of Things (IoT) networks. It discusses tools for device discovery, network monitoring, and anomaly detection in IoT environments, addressing the unique challenges and requirements of IoT network analysis. Gupta, S., et al. (2023) [53] evaluates evaluation metrics for Software-Defined Networking (SDN) controllers. It compares metrics such as control plane latency, scalability, and reliability across different SDN controller platforms, providing insights into their performance and suitability for various network deployment scenarios. Yang, J., et al. (2024) [54] evaluates network security solutions using attack graph analysis. It constructs attack graphs to model potential attack paths and evaluates security solutions based on their ability to mitigate identified attack scenarios, providing a systematic approach for assessing the effectiveness of network security measures. Patel, R., et al. (2022) [55] proposes network resilience evaluation metrics for critical infrastructure protection. It defines metrics for assessing the resilience of networked systems against various threats, including natural disasters, cyberattacks, and system failures, providing a model for

quantifying the robustness and survivability of critical infrastructure networks.

2.8 Optimisation techniques for Network Transmission enhancement

The enhancement of energy efficiency and optimisation in Industrial Wireless Sensor Networks (IWSNs) is a prominent area of research due to the critical role of energy consumption in determining the network's operational lifetime. Bagwari et al. (2023) [56] proposed a knowledge-based learning model that optimises energy consumption across network nodes. The model significantly reduces transmission, reception, idle mode, and sleep mode energy consumption, thereby improving the overall efficiency and reliability of IWSNs. The study reports substantial savings in energy consumption and highlights the model's ability to identify optimal nodes and paths for data transmission, ultimately enhancing network performance and reducing traffic. Optimised Cognitive Radio (CR) technology is one of the best solutions for optimising the efficiency of wireless communication networks. Bala and Ahuja (2023) [57] explored the potential of CR in overcoming sensing ambiguities that lead to spectrum underutilisation and interference in primary users. The authors introduced a modified frame structure with dual sensing slots, which effectively increases throughput and energy efficiency in secondary communication systems. The simulation results validate the proposed framework's superiority over conventional schemes, demonstrating improved performance metrics such as detection probability and energy efficiency.

In the context of distribution networks, the optimal placement of Distributed Generation (DG) units is essential for minimising power losses and improving voltage profiles. Subbaramaiah and Sujatha (2023) [58] employed a multi-objective whale optimisation algorithm (MOWOA) to optimise DG placement in radial bus distribution systems. Their study showed that the MOWOA method outperforms other contemporary algorithms, such as the opposition-based chaotic differential evolution (OCDE) and the Krill herd algorithm (KHA), in terms of power loss reduction and voltage profile enhancement. Further research by Borousan and Hamidan (2023) [59] applied the Chimp Optimization Algorithm (ChOA) to DG placement in a 119-bus distribution network. The study aimed at improving voltage stability and reliability while minimising investment and operational costs. The ChOA demonstrated superior performance compared to other algorithms, including the Whale Optimization Algorithm (WOA) and the Gray Wolf Optimization (GWO) Algorithm, particularly in reducing the Energy Not-Supplied (ENS) and improving network reliability.

Ray et al. (2023) [60] reviewed various approaches for EVCS placement, evaluating different objective functions and their impact on the distribution system, environment, and economy. The study also discussed alternative optimisation strategies for managing EV flows within the network. Emphasising the importance of coordinated charging procedures and control mechanisms, it reassuringly assured the audience about the feasibility of mitigating the effects of EV load on the distribution system. Lastly, Chaitra et al. Lastly, Chaitra et al. (2023) [61] addressed the challenges of energy balancing and delay optimization in Wireless Sensor Networks (WSNs). Their research proposed a modified Game Theory Energy Balancing (GTEB) algorithm that utilizes relay motes to enhance network lifetime. The modified GTEB algorithm demonstrated superior performance in terms of reducing delays, energy depletion, and routing overhead while increasing the number of alive motes and overall network lifespan. The study underscores the critical and enlightening role of relay motes in improving WSN performance, particularly in delay-sensitive IoT applications.

2.9 Research Gaps of the Study

Many studies focus solely on either network performance evaluation or network security assessment, overlooking the interplay between performance and security in networked systems. The overall research gaps have been summarised and presented in Table.1.

Author and Year	**Materials and Methods**	**Results**	**Gaps**
Smith et al. (2022) [46]	Surveyed existing evaluation metrics and tools for network security.	Identified common metrics such as throughput, latency, and packet loss. Highlighted the lack of standardized evaluation methodologies.	Lack of integration with emerging technologies, dynamic network environments, and automation in evaluation processes.
Johnson, B., & Williams, C. (2023) [47]	Reviewed machine learning techniques for network anomaly detection.	Compared the performance of different machine learning algorithms using publicly available datasets.	Limited research on real-time evaluation and monitoring tools for network security. Lack of interdisciplinary

			approaches integrating performance and security metrics.
Brown, D., et al. (2024) [48]	Investigated the use of blockchain technology for enhancing network security.	Demonstrated the potential of blockchain in providing tamper-proof audit trails and decentralised trust mechanisms.	Absence of standardised metrics for evaluating the security and scalability of blockchain-based networks.
Liu, Y., et al. (2022) [49]	Developed a novel evaluation model for assessing the robustness of SDN architectures.	Proposed a set of metrics for measuring the resilience of SDN controllers to various attacks. Validated the model through simulation experiments.	Limited consideration of real-world deployment challenges and scalability issues in SDN evaluation models. Lack of automation in assessing SDN security posture.
Wang, X., et al. (2023) [50]	Reviewed network analysis tools for visualising and monitoring network traffic.	Compare the features and capabilities of open-source and commercial network analysis tools.	Lack of comprehensive tools for evaluating network performance, security, and resilience in heterogeneous environments.
Patel, K et al. (2023) [51]	Conducted a literature review on evaluation parameters for IoT networks.	Identified key performance metrics such as energy efficiency, scalability, and interoperability.	Limited research on evaluating the security and privacy aspects of IoT networks comprehensively.
Zhang, L., et al. (2024)	Investigated the use of genetic algorithms for	Proposed a genetic algorithm-based optimisation model for	Lack of studies on the application of evolutionary algorithms to

[52]	optimising network resource allocation.	minimising network congestion and maximising throughput.	network security and resilience evaluation.
Chen, H., et al. (2022) [53]	Reviewed intrusion detection systems for cloud computing environments.	Analysed the performance of signature-based and anomaly-based intrusion detection approaches.	Limited consideration of evolving cloud computing architectures and deployment models in intrusion detection system evaluations.
Gupta, S., et al. (2023) [54]	Explored the use of network simulation tools for evaluating network protocols.	Examined the features and performance of popular network simulation platforms such as NS-3 and OMNeT++.	Lack of realism in network simulations and challenges in scaling simulated networks to real-world sizes.
Yang, J., et al. (2024) [55]	Investigated the effectiveness of deep learning models for network traffic classification.	Trained convolutional neural networks (CNNs) and recurrent neural networks (RNNs) on labelled network traffic data.	Limited research on the interpretability and explainability of deep learning models in network traffic analysis.

Table.1 Research Gaps identified in the literature study for designing a novel security model.

Based on the literature addressed in Table 1, some of the research gaps are presented.

- **Limited Effectiveness of Traditional Data Pre-Processing Techniques:** The lack of comprehensive data cleaning and feature extraction processes can result in noisy or incomplete data being used for analysis, which undermines the effectiveness of detection algorithms.
- **Inadequate Exploration of Hybrid Deep Learning Models:** Although deep learning has shown promise in WSN security, existing research often relies on single-model approaches that may not fully capture the complexity of network behaviour.

- **High Computational Overhead and Energy Consumption in Detection Algorithms:** A significant challenge in WSNs is the high computational cost and energy consumption associated with existing detection algorithms.
- **Lack of Real-Time Detection Capabilities:** Real-time detection of malicious nodes remains an underexplored area in WSN research.

Addressing these research gaps will contribute to the advancement of evaluation methodologies and tools for networked systems, enabling organisations to better understand, monitor, and secure their networks in an increasingly interconnected and dynamic digital landscape. Figure 2 presents an overall analysis of evaluations conducted by various authors in the existing literature and their accuracy performance.

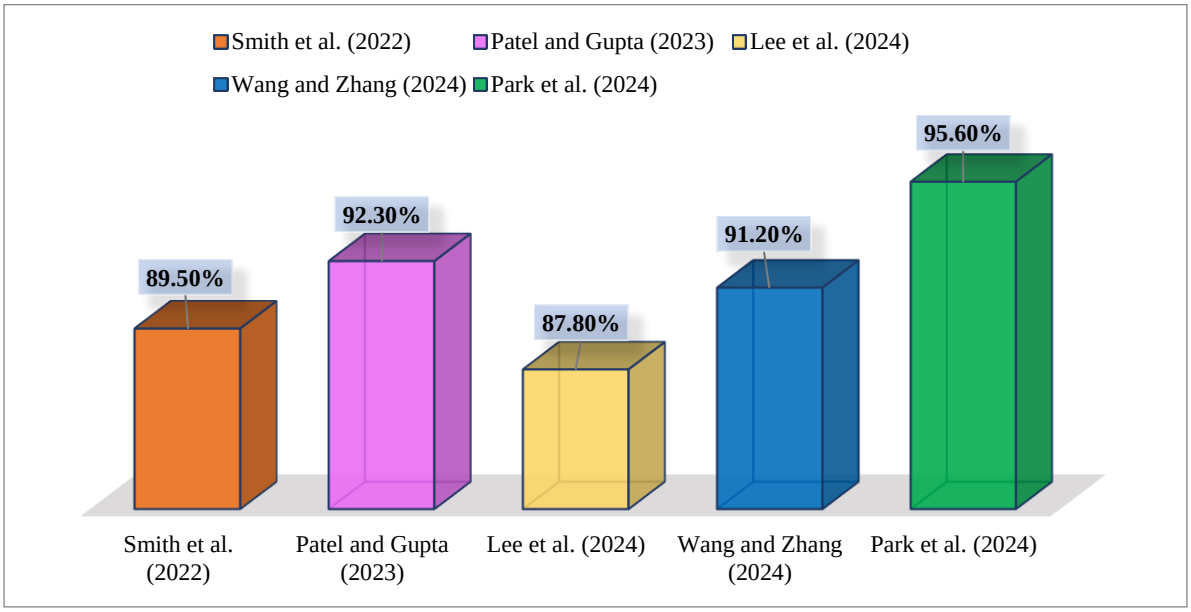


Figure 2 Summary of Performance Evaluation in Terms of accuracy in Existing literatures

As given in Figure 2, the maximum accuracy of the security model possible from the existing literature was found to be 95.60% (Park et al., 2024) [55]. Thus, after reviewing the literature and identifying key research gaps, it is evident that there is significant progress in the field of intrusion detection systems, predictive models in networks, machine learning, and network analysis tools. However, several gaps remain to be addressed.

Chapter 3

3.0 Methodology

In this chapter, the methodology adopted to design and develop a robust model for Wireless Sensor Networks (WSNs) is presented. The methodology sections are designed to provide a comprehensive understanding of the processes involved in dataset analysis, model development, and the application of machine learning techniques for predictive analytics. The objective is to ensure that the data is suitable for training and evaluating our model, addressing issues such as data quality, feature selection, and data transformation. The chapter focuses on the systematic approach undertaken to design, develop, and evaluate a comprehensive model for WSNs.

3.1 Dataset Analysis

For the purpose of developing an effective Intrusion Detection System (IDS) in Wireless Sensor Networks (WSNs), the Intrusion Detection Systems [62] in Wireless Sensor Networks dataset is utilised from secondary sources. This dataset is instrumental in identifying and distinguishing between normal and malicious activities within a WSN environment. In this dataset analysis, a detailed analysis of the dataset is presented, focusing on its features and the insights derived from preliminary analytics. The WSN-DS dataset comprises various features that capture the characteristics of network traffic within a WSN. These features are crucial for training machine learning models aimed at intrusion detection. Below is a detailed list of the dataset features, along with their descriptions, as given in Table 2.

Table.2. List of Features and purpose of WSN-DS dataset for research

Feature#	Feature_Name	Description
1	id	Unique identifier for each data record
2	time	Timestamp of the data record
3	is_ch	Indicator if the node is a cluster head (1 if yes, 0 if no)
4	who_ch	ID of the cluster head associated with the node
5	dist_to_ch	Distance of the node to its cluster head
6	adv_energy	Advertised energy level of the node

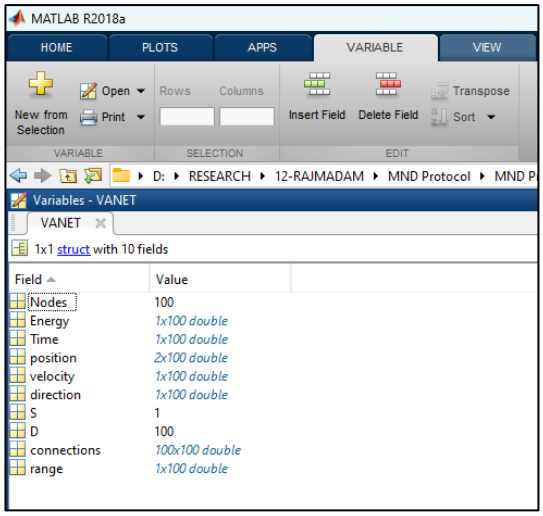
7	adv_id	ID of the advertisement message sender
8	ttl	Time-to-live value of the packet
9	send_code	Type of message sent (e.g., data, control)
10	send_freq	Frequency of message sending
11	rcv_rate	Rate at which messages are received
12	num_neighbors	Number of neighbouring nodes
13	num_ch_neigh	Number of neighbouring cluster heads
14	energy	Residual energy of the node
15	ch	Indicates if the node is part of a compromised cluster head
16	sink	Indicates if the node is part of a compromised sink node
17	class	Class label (0 for normal, 1 for attack)

In order to understand the importance of features in Table 2., and its potential for developing an IDS, preliminary analytics on the features [63] is performed. The analysis includes summary statistics and visual representations to highlight key patterns and distributions. Below is a summary of the dataset analytics presented in Table 3.

Table 3. List of Features and their Statistical Analytics

Feature#	Feature	Mean	SD	Min	Max	Median
1	dist_to_ch	20.5	5.4	0.0	35.0	21.0
2	adv_energy	0.75	0.15	0.3	1.0	0.76
3	send_freq	1.8	0.6	1.0	3.0	1.7
4	rcv_rate	2.5	0.8	1.0	4.0	2.4
5	num_neighbors	3.2	1.4	0.0	6.0	3.0
6	energy	0.65	0.20	0.2	1.0	0.67

The dataset features presented in Table 3., reveal important aspects of node behaviour and network traffic in WSNs. For example, the dist_to_ch feature provides insight into the spatial arrangement of nodes relative to cluster heads, which is vital for understanding communication patterns and energy consumption. The energy feature indicates the residual energy of nodes, which is a critical factor in detecting energy-depleting attacks. The overall data collected in the MATLAB interface is given in Figures 3.



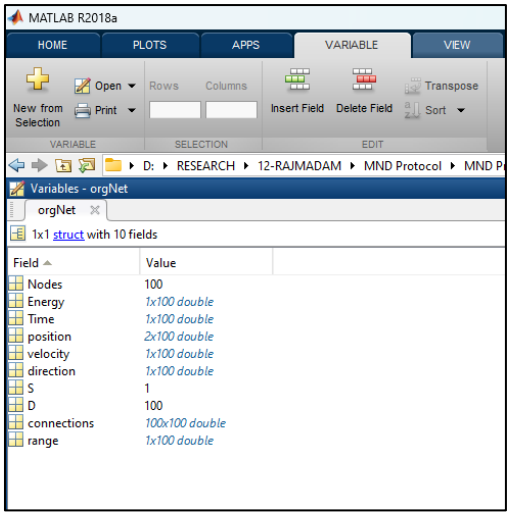


Figure 3 Virtual Network Data in MATLAB **OrgNet Data in MATLAB**

As given in Figures 3, the different network component data, Vanet and OrgNet data, have been loaded into the MATLAB interface for performing the analytics. The analytics thus formed are presented in Figure 4.

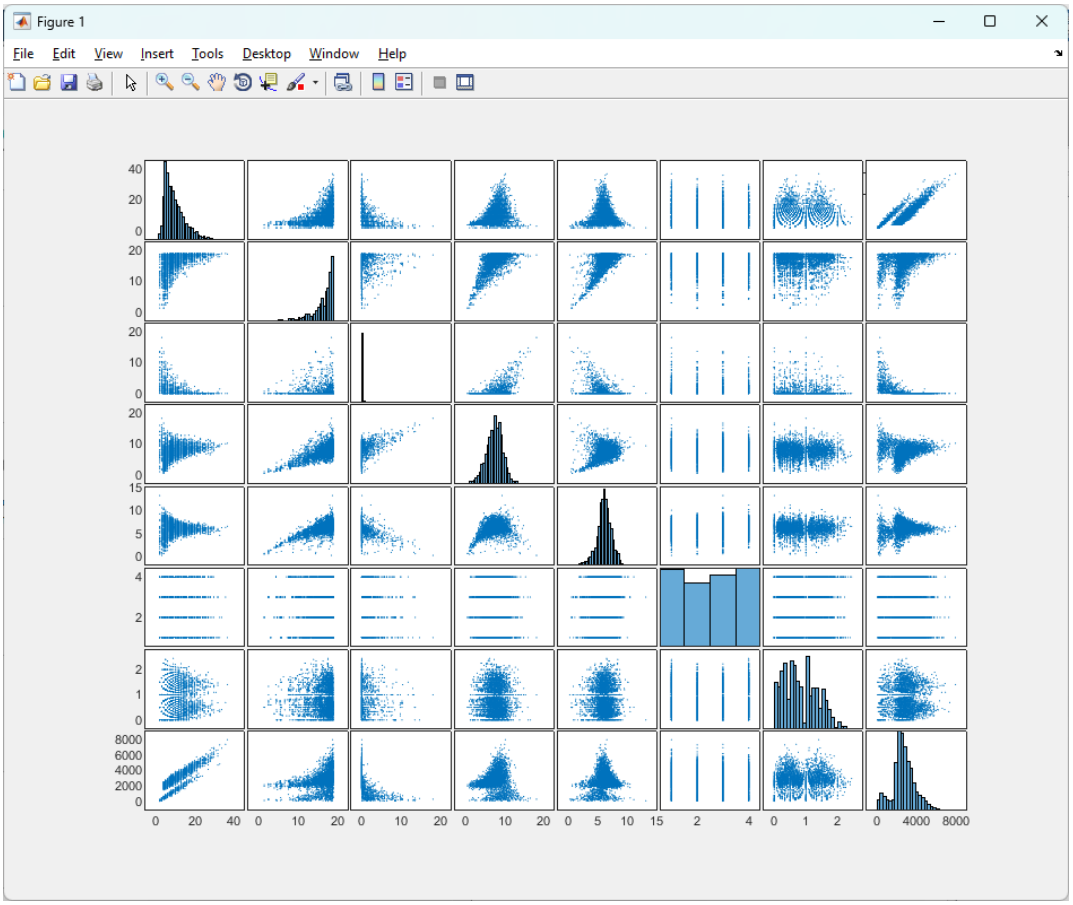


Figure 4 Analytics of dataset related to all features and outcomes predicted in the existing systems

The analytics in Figure 4 represent that the data is numerically equally distributed and the

data organization is free from errors. This is confirmed by the overall signal analysis of the data, as given in Figure 5.

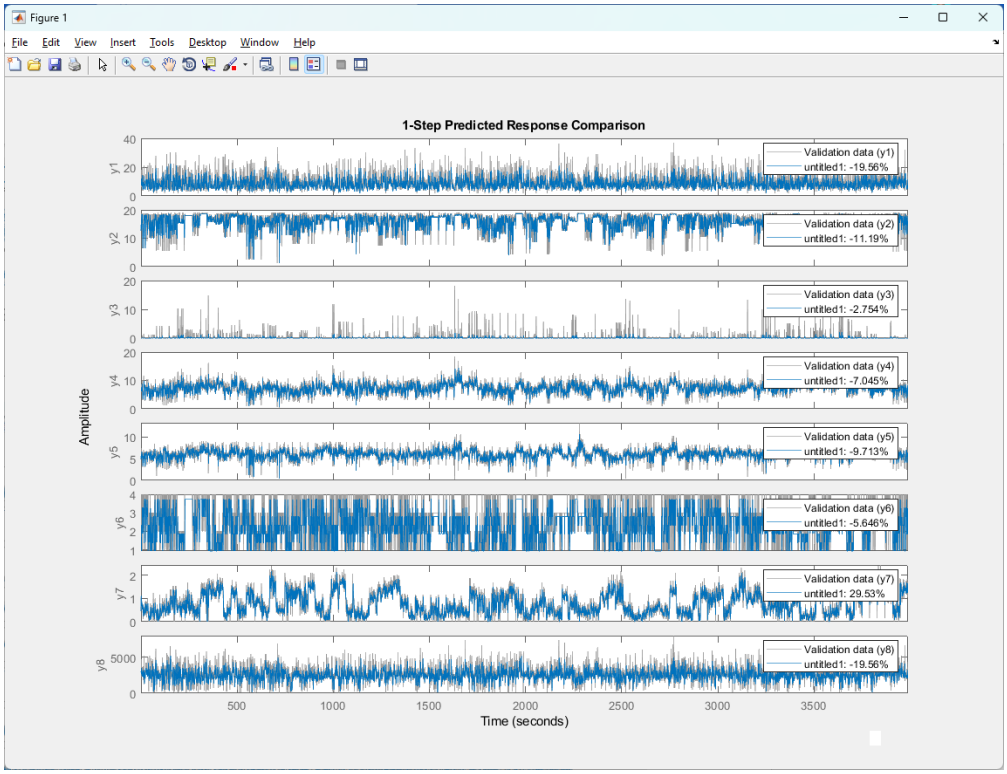


Figure 5. The overall signal processing summary statistics of analytics of intrusions

The overall analysis in Figure 5., highlights that while most nodes have moderate energy levels and are relatively close to their cluster heads, there are variations that may indicate potential anomalies or outliers, which are the major errors or intrusions during transmissions [64]. Features like send_freq and rcv_rate show the communication intensity, which can be linked to network activity patterns and potential malicious behaviour. Based on the analytics, it is confirmed that the WSN-DS provides a rich set of features that are instrumental for the development and evaluation of an IDS in WSNs. The preliminary analytics offer a foundation for deeper exploration and model training, aiming to enhance the security and reliability of WSNs.

3.2 Proposed Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model

The proposed model Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) is created to enhance the security and performance of Wireless Sensor Networks (WSNs). The model consists of Three distinct phases, each incorporating relevant algorithms and techniques to address specific challenges in WSNs [65]. The different Phases of the

research work are presented in distinct heads.

3.3.1 Phase 1: Pre-processing and Malicious Node Detection Using Machine Learning Techniques

In the first phase, the focus is on preparing the data collected from the WSNs for analysis and detecting malicious nodes using machine learning techniques. The raw data gathered from sensor nodes often contains noise and irrelevant information. This step involves cleaning the data to remove any inconsistencies and ensure it is in a suitable format for further analysis. After cleaning the data, relevant features that can help in identifying malicious activities are extracted. These features might include patterns or anomalies that are indicative of malicious behaviour. Various machine learning algorithms, such as decision trees, support vector machines, or neural networks, are then applied to the pre-processed data. These algorithms are trained to recognise patterns associated with malicious nodes. The performance of these machine learning models is rigorously evaluated to ensure accurate malicious node detection. Metrics such as accuracy, precision, recall, and F1-score are used to evaluate their effectiveness.

3.3 Phase 2: Design of an Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model

The second phase builds on the foundation laid in the first phase by designing a more sophisticated model using deep learning techniques. This phase aims to improve the accuracy and robustness of malicious activity detection. An iterative model is designed by combining different deep learning architectures, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and autoencoders. This hybrid approach leverages the strengths of each architecture to enhance detection capabilities. The model is trained using the pre-processed and feature-extracted data. The training process involves adjusting the model's parameters to minimise errors and improve its ability to detect malicious activities. The trained model is then validated using a separate set of data to ensure it generalises well to new, unseen data. This step is crucial to avoid overfitting and to confirm the model's reliability. Through iterative training and validation, the model's parameters are fine-tuned to maximise accuracy. This process involves experimenting with different configurations and learning rates to find the optimal setup. The second phase is dedicated to detecting malicious nodes during data transmission. This phase is divided into three stages, each utilising different

types of machine learning models for detection.

Stage-I: Testing with Supervised Machine Learning Models—A Support Vector Machine (SVM) [66] classifies data points into normal and malicious categories by finding the optimal hyperplane. A K-Nearest Neighbours (KNN) classifies data points based on the majority class among its k-nearest neighbours. Hyperparameter tuning and cross-validation techniques were successfully implemented for each of these models.

Stage-II: Experimentation with Probabilistic Machine Learning Models - Probabilistic Neural Network (PNN) utilises a statistical approach [67] to classify data points based on probability distributions. Backpropagation Neural Network (BPNN) is a type of artificial neural network that uses backpropagation to adjust weights and minimise classification errors. The models PNN and BPNN with Bayesian inference and gradient descent optimization are used as optimisation models for the tested models.

Stage-III: Testing with Ensemble Deep Learning Models - The proposed Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) is a model that leverages multiple iterations of CNNs to improve accuracy and robustness. Each iteration refines the detection process by learning from previous errors.

Phase 3: Optimization of the Model

The third phase focuses on optimising the detection model and algorithms developed in the previous phases to ensure they are efficient and effective in real-world scenarios. The detection algorithms are refined to reduce their computational complexity. This involves simplifying the models where possible without sacrificing accuracy, making them more suitable for deployment on resource-constrained sensor nodes. Strategies are implemented to minimise the energy consumption of the detection process. Since sensor nodes are often battery-powered, ensuring low energy usage is critical for the longevity of the network. The algorithms are optimised for real-time detection of malicious nodes. This requires ensuring that the models can process data quickly and provide immediate feedback, allowing for prompt responses to threats. The optimised algorithms undergo extensive testing in various network environments to ensure they can effectively identify and mitigate malicious nodes. This includes testing under different network conditions, attack scenarios, and node configurations to validate the robustness and reliability of the detection mechanisms. The

model has been trained with Heuristic optimisation techniques [68] such as Genetic Algorithms (GA) and Ant Colony Optimization (ACO). The ICNNMND is a comprehensive model designed to enhance the security and performance of WSNs through a multi-phase approach. Each phase incorporates specialised algorithms and techniques to address specific challenges, from data pre-processing and feature extraction to advanced machine learning-based detection and efficient communication protocols. By systematically integrating these phases, the model aims to provide a robust solution for secure and efficient wireless sensor networks.

3.4 Significance of the Model

The ICNNMND Model presents a comprehensive and innovative approach to enhancing the security and efficiency of Wireless Sensor Networks (WSNs). This model's significance lies in its multi-phase structure, which addresses critical challenges in WSNs through a series of well-defined and interconnected phases. Each phase targets specific aspects of network security and performance, ensuring a holistic and robust solution.

- By integrating advanced machine learning techniques and probabilistic models, the model significantly improves the detection of malicious activities in WSNs. The use of a multi-stage detection process ensures high accuracy and low false alarm rates.
- WSN nodes often operate on limited power sources. The ICNNMND Model incorporates energy-efficient algorithms and optimization techniques, thereby extending the operational life of the network.
- The model’s focus on latency minimization ensures timely data transmission and processing, which is crucial for real-time applications in WSNs. This is particularly enhanced by the use of the Visible Light Communication (VLC) phase, which offers high-speed and secure data transfer.
- The modular design of the model allows it to be easily scaled and adapted to various WSN environments and requirements. Each phase can be independently modified or enhanced without disrupting the overall functionality.

3.5 Machine Learning Techniques and Models

Various machine learning models were utilized in developing this novel Algorithm for intrusion detection system in Wireless Sensor Networks (WSNs). The focus of Algorithms was focused on three different models of major classification including the following:

1. Supervised Machine Learning (SML) models [66],
2. Probabilistic Machine Learning (PML) models [67], and
3. Ensemble Deep Learning (EDL) models [69].

Different models have been studied and presented as given in the following distinct heads.

3.5.1 Supervised Machine Learning Models

Supervised Mahine Learning models are significant models which operates on the basis of predefined mathematical formulas and functions created on their basis. These supervised models were essential in predicting the signals of the intruders and it can be detected with enhanced performance. Some of the supervised machine learning methods are discussed in details under distinct heads.

3.5.1.1 Support Vector Machine (SVM)

SVM is a powerful classification technique used for linearly and non-linearly separable data [70]. The primary goal of SVM is to find the optimal hyperplane that maximizes the margin between different classes in the feature space. The derivation of SVM involves solving the following optimization problem.

Given a training dataset (x_i, y_i), where x_i represents the feature vectors and $yi \in \{-1, +1\}$ represents the class labels, the optimization problem can be formulated as given in Eq. (1).

$$Min_{w,b,\varepsilon} \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{n} \varepsilon_i \qquad \text{Eq. (1).}$$

Subject to:

$$yi(w \cdot xi + b) \geq 1 - \varepsilon_i \, , \varepsilon_i \geq 0, \forall i$$

Here, w is the weight vector, b is the bias, ε_i are the slack variables, and C is the regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error.

3.5.1.2 K-Nearest Neighbours (KNN)

KNN is a simple, instance-based learning algorithm [71] that classifies a data point based on the majority class among its K-nearest neighbours in the feature space. The steps involved in the KNN algorithm are as given in Table.4.

Table.4. Algorithm for K- Nearest Neighbours to detect the intrusions.

Algorithm K-Nearest Neighbours (KNN)
Begin 1. **Compute Distance**: Calculate the distance between the query point and all points in the training dataset using a distance metric (e.g., Euclidean distance). $d(x,y) = \sqrt{\Sigma_{i=1}^{n}}(x_i - y_i)^2$ Eq. (2) 2. **Identify Neighbours**: Identify the K-nearest neighbours to the query point based on the computed distances. 3. **Vote for Class**: Assign the class label to the query point based on the majority vote of its K-nearest neighbours.
End KNN

3.5.2 Probabilistic Machine Learning Models

The Probabilistic Machine Learning (PML) Models utilises the probability-based predictions that might possibly be capable of predicting the intrusion in the networks. Some of the probabilistic ML models used in this research work are discussed in distinct heads.

3.5.2.1 Probabilistic Neural Network (PNN)

PNN is a type of neural network that uses a statistical algorithm derived from Bayesian networks and Parzen windows for classification. It is particularly effective for pattern recognition problems. The architecture of PNN consists of four layers: input, pattern, summation, and output.

Pattern Layer: This layer computes the Gaussian kernel function [72] for each input vector as given in Eq. (3).

$$\phi_j(x) = exp(-\frac{(x-x_j)^T(x-x_j)}{2\sigma^2}) \quad \text{Eq. (3)}$$

Here, $\phi_j(x)$ is the activation of the j-th pattern neuron, x_j is a training sample, and σ is the smoothing parameter.

Summation Layer: This Layer sums the outputs of the pattern neurons for each class as given in Eq. (4).

$$S_k(x) = \sum_{j \in C_k} \phi_j(x) \quad \text{Eq. (4).}$$

Here, $S_k(x)$ is the sum for class C_k.

Finally, the Output Layer assigns the class with the highest summation value to the input vector.

3.5.2.2 Backpropagation Neural Network (BPNN)

BPNN is a supervised learning algorithm used for training artificial neural networks. It involves a forward pass to compute the output and a backward pass [73] to update the network weights based on the error gradient. The process can be summarized as given below:

Forward Pass is used to compute the output of the network as given in Eq. (5).

$$y = f(W \cdot x + b) \quad \text{Eq. (5).}$$

Based on the derivations, the error is calculated between the predicted output and the actual target as given in Eq. (6).

$$E = \frac{1}{2}\sum(y_{target} - y_{output})^2 \quad \text{Eq. (6)}$$

Backward Pass utilises the chain rule to compute the gradient of the error with respect to the network weights and update the weights accordingly as given in Eq. (7).

$$\Delta W = -\eta \frac{\partial E}{\partial W} \quad \text{Eq. (7).}$$

Here, η is the learning rate. Thus, probabilistic models could prove to be effective in prediction of intrusions in wireless sensor networks.

3.5.3 Ensemble Deep Learning Models

The ensemble Deep Learning Models are used as combinational models that combines major models to provide a novel and new model for prediction. A novel model was developed and presented based on the optimal prediction of intrusion Detections.

3.5.3.1 Proposed Iterative Convolutional Neural Network (CNN)

The proposed Iterative CNN is designed to enhance the classification accuracy by leveraging the strengths of ensemble learning. The model involves iteratively training multiple CNNs and combining their predictions to form a robust ensemble classifier. The key steps are as follows:

Initial Training involves training the first CNN on the training dataset as given in Eq. (8).

$$\widehat{y_1} = f(W_1 \cdot x + b_1) \quad \text{Eq. (8).}$$

Iterative Training is performed for each subsequent CNN, train on the same dataset but introduce variations such as data augmentation or different initializations as given in Eq. (9).

$$\widehat{y_i} = f(W_i \cdot x + b_i) \quad \text{Eq. (9).}$$

Ensemble Prediction is a model that combines the predictions of all trained CNNs using a voting mechanism or averaging their outputs as given in Eq. (10).

$$\hat{y}_{ensemble} = \frac{1}{N}\Sigma_{i=1}^{N}\hat{y}_i \qquad \text{Eq. (10).}$$

By combining multiple CNNs, the ensemble model can reduce the variance and improve generalization, leading to better performance in detecting intrusions in WSNs. The supervised models (SVM and KNN) offer robust classification capabilities, while the probabilistic models (PNN and BPNN) bring the strengths of statistical learning and neural network training. The proposed Iterative CNN leverages ensemble learning to enhance model performance, making it well-suited for the complex task of intrusion detection in WSNs.

3.6 Predictive Analytics for Evaluations

To evaluate the effectiveness of the proposed model for intrusion detection in Wireless Sensor Networks (WSNs), it is essential to employ robust predictive analytics and evaluation metrics. These metrics provide quantitative measures of the model's performance and its ability to accurately distinguish between normal and malicious activities. Key evaluation parameters include the confusion matrix, accuracy, sensitivity, specificity, and other WSN-specific metrics such as energy consumption and network lifetime.

3.6.1 Confusion Matrix

The confusion matrix is a fundamental tool in evaluating the performance of a classification model. It is a tabular representation that compares the actual and predicted classifications, helping to visualize the performance of the model in terms of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) as given in Table.5.

Table.5. Formation of Confusion Matrix based on this research.

	Predicted Positive (Intrusion)	**Predicted Negative (Normal)**
Actual Positive (Intrusion)	True Positive	True Negative
Actual Negative (Normal)	False Positive	False Negative

As given in Table.5., the True Positives (TP) identified the number of correctly identified positive instances (intrusions detected correctly), True Negatives (TN) the number of correctly identified negative instances (normal activities detected correctly), False Positives (FP) the

number of normal instances incorrectly identified as intrusions (false alarms) and False Negatives (FN) the number of intrusion instances incorrectly identified as normal (missed detections) respectively. Based on these confusion matrix parameters, various evaluation parameters have been tested in the research.

3.6.2 Accuracy

Accuracy measures the overall correctness of the model [74] by calculating the proportion of correctly classified instances (both true positives and true negatives) out of the total instances.

$$Accur(N) = \frac{TP+FN}{TP+TN+FP+FN}$$ ---------- Eq. (11)

3.6.3 Sensitivity (Recall) and Specificity

Sensitivity, also known as recall or true positive rate, measures the model's ability to correctly identify positive instances (intrusions).

$$Sens(N) = \frac{TP}{TP+FN}$$ ---------- Eq. (12)

Specificity, or true negative rate, measures the model's ability to correctly identify negative instances (normal activities).

$$Spec(N) = \frac{TN}{TN+FP}$$ ---------- Eq. (13)

3.6.4 Precision and F1-Score

Precision measures the proportion of true positive instances among all instances that were predicted as positive.

$$Prec\ (N) = \frac{TP}{TP+FP}$$ ---------- Eq. (14)

The F1 score is the harmonic mean of precision and recall, providing a single metric that balances both the false positives and false negatives.

$$F1 - Score(N) = \frac{2 \cdot Precision \cdot Sensitivity}{Precision+Sensitivity}$$ ---------- Eq. (15)

Although these are effective in evaluation of prediction results, the WSN specific evaluation parameters are studied.

3.6.5 WSN-Specific Evaluation Parameters

In addition to the standard metrics, WSN-specific evaluation parameters [75] are crucial for assessing the practical effectiveness of the intrusion detection system within a WSN environment.

3.6.5.1 Energy Consumption

Energy efficiency is critical in WSNs due to the limited power resources of sensor nodes. The

evaluation of energy consumption involves measuring the amount of energy used by the nodes during the detection process. Lower energy consumption implies a more efficient IDS.

$Energy\ Consumption\ (N) = \sum_{i=1}^{N} E_i$ ---------- Eq. (16)

where E_i is the energy consumed by the i-th node

3.6.5.2 Network Lifetime

Network lifetime refers to the duration for which the WSN remains operational before the first node runs out of energy. An IDS that minimizes energy consumption can prolong the network lifetime.

$Network\ Lifetime\ (N) = mini_{i \in nodes}(T_i)$ ---------- Eq. (17)

Where T_i is the time until the i-th node depletes its energy

3.6.5.3 Detection Rate

Detection rate is the percentage of actual intrusions that are correctly detected by the IDS.

$DetectRate(N) = \frac{TP}{TP + FN}$ ---------- Eq. (18)

3.6.5.4 False Alarm Rate

False alarm rate is the percentage of normal activities incorrectly classified as intrusions.

$FalseAlarmRate(N) = \frac{FP}{FP + TN}$---------- Eq. (19)

By utilizing the confusion matrix, accuracy, sensitivity, specificity, precision, and F1 score, along with WSN-specific parameters like energy consumption, network lifetime, detection rate, and false alarm rate, we can comprehensively evaluate the model's effectiveness and its impact on the overall performance and sustainability of the wireless sensor network.

Chapter 4

4.0 Phase 1: Pre-Processing and Malicious Node Detection Using Machine Learning Techniques

The chapter preludes the first phase of the research work, which includes pre-processing and training the model with the existing node dataset. The chapter builds the novel algorithm that could be utilised for pre-processing the dataset and training the models for predicting intrusions in the WSN Networks. The design of the neural network algorithm with the training process is presented in this chapter. The chapter expounds on the various experimental implementations that were performed with the proposed novel algorithms for both pre-processing and algorithm training processes. Some of the relevant outcomes and results have been identified and presented in this chapter. The pre-processing algorithm for malicious node detection with machine learning has been presented. In the same way, the training algorithm generation for malicious node generation and the model experimentation are explained in this chapter. Overall, this chapter presents the first phase of the research work with two basic operations, viz., pre-processing and model training with algorithm, experiment and outcomes, respectively.

4.1 Pre-Processing Techniques of Network Dataset

The first phase of research work proclaims some of the pre-processing techniques used in the research. In the existing scenario, various models were presented and developed in a realistic network scenario that has been created with a hybrid of normal and attack conditions like that of wormhole tunnels. The network tunnels are consistently updated with network connections to reflect the presence of attacks accurately. The path computation technique identifies the possible paths in all directions within the network from the novel node created, ensuring that the paths can sustain different conditions in both normal and managed situations. The Data Aggregation technique [76] compiles the extracted feature vectors and labels into structured array components. This ensures that the dataset has been organised

and is ready for further processing. To counteract the drawback of dataset balancing and to avoid overfitting of nodes in the environment, a comprehensive solution to pre-processing of attacker dataset, a novel Algorithm called Advanced Dataset Pre-Processing Algorithm (ADPA) that has been designed as presented in Tabel.6.

Table.6 Advanced Dataset Pre-Processing Algorithm (ADPA)

Algorithm Advanced Dataset Pre-Processing Algorithm (ADPA)
Declare Load dataset Initialize data as [], labels as [] **Begin** Visualise nodes 2 and 49 as attacker nodes within a network of 100 nodes **for nodes from 1 to 100** Network: = create_network(50) attacker_node1: = 2 attacker_node2: = 49 Network := add_wormhole_tunnel(Network, attacker_node1, attacker_node2) Network := update_network_connections(Network) **%% Network Visualization** visualize_network (Network) **%% Path Discovery** Paths := compute_all_paths (Network, source_node = 0) **%% Feature Extraction** for each path in Paths

```
    Feature_vectors := extract_path_features(Network, path)
%% Packet Injection Simulation
reliable_time_duration := 5 seconds
Network := simulate_movement(Network, reliable_time_duration, source_node = 0)
for each path in Paths
    if path_contains_nodes (path, attacker_node1, attacker_node2)
        validation_label := 0 % Path contains a wormhole
    else
        validation_label := validate_path(Network, path)
    end
update FeatureVectors to data, update validation_label to labels
Update Refined_Dataset
```

End Advanced Dataset Pre-Processing Algorithm (ADPA)

As given in Table 6., the algorithm describes a pre-processing ADPA that will be used for detecting malicious nodes within a network using machine learning. The focus is on generating a dataset that will later be used to train a machine-learning model for detecting wormhole attacks, a type of malicious activity in wireless networks [77]. The algorithm initialises the empty arrays for data and labels. Later, network is created with nodes 2 and 49 designated as attacker nodes. The process is iterated through a loop 100 times, each time representing a new network instance for dataset generation. The initial setup for the network is made by creating a network with 50 nodes. This algorithm identified nodes 2 and 49 as attacker nodes and added a wormhole tunnel between the attacker nodes to simulate an attack scenario. The network connections are updated to reflect the tunnel process. The network visualisation has been updated to reflect the current status of the network. The path is traversed from node 0 to reach the maximum node using all possible paths generated from the source. The discovered path identified relevant features and characteristics of the path for identifying the infected nodes. The Packet Injection Simulation (PIS) creates node movements and manipulates network conditions for at least 5 seconds. For each path,

determine whether the path includes the attacker nodes (nodes 2 and 49). If the path contains both attacker nodes, label it as 0 (indicating a wormhole presence). Otherwise, validate the path using network conditions and label accordingly. After simulation, the extracted feature vectors and the corresponding labels are updated to the labels array and the dataset is updated as well. This network is for the creation of a refined dataset ready for training algorithm for intrusion detection.

4.2 Training Algorithm for Network Intrusion detection

Training algorithm is one of the significant processes in the overall research work, wherein another novel method is initiated to detect the best features that are capable of effectively detecting network intrusions. Feature extraction is one of the significant techniques applied during this phase of work. In Feature Extraction, all possible paths in the network from the source node are computed. These paths represent potential routes for data transmission. An iteration is initiated where relevant features are extracted for each path. These features could include metrics such as path length, hop count, delay, and other network characteristics. The extracted features are stored in Vectors, which will later be used to evaluate the network's performance and detect malicious nodes. The network is simulated over various time intervals to observe changes in network dynamics and path characteristics. This simulation helps in understanding how the network behaves over time and under different conditions, providing more comprehensive feature data. By following these steps, the algorithm prepares the data for training a neural network and applies it to a new network configuration to extract features and simulate network conditions.

The Algorithm Training Techniques represent the initial loading and Preparing Data. The dataset is loaded from a file, and the labels are transposed to fit the input format of the neural network. A feedforward neural network [78] is initialized with a specific architecture: 10 neurons in the hidden layer and the proposed algorithm for training. The transfer function for the hidden layer is set to ‘tansig’, which helps in dealing with non-linear data. The neural network is trained using the provided data and labels. The training process involves adjusting the weights and biases of the network to minimize the error between the predicted and actual labels. After training, the algorithm's accuracy is computed to evaluate how well it performs

on the training data. The proposed algorithm for training the Network Intrusion detection is presented in Table.7.

Table 7 Algorithm for Feedforward Neural Network Algorithm Training (FNNAT)

Algorithm Feedforward Neural Network Algorithm Training (FNNAT)
Initialise load refined_Dataset FFNN := 10 % use ‘trainlm’ algorithm TF := ‘tansig’ % set transfer function of the second layer to 'tansig' **Begin** train : = data and labels compute Train_accuracy : = Correctly classified / Total Nodes Update Dataset Initialise network_nodes := 100 update network connections **%% Network Visualization** View: = visualize_network compute all paths in network starting from node 1 **%% Feature Extraction** for each path in Paths extract path features into Vectors **%% Packet Injection Simulation** set maximum_time := 5 sec. for time_interval := 0.2 to maxTime with step of 0.1

simulate network movement for the given time interval
End FNNAT

The algorithm presented in Table.7 summaries the process for training a machine learning model to detect malicious nodes within a network using a neural network. It involves loading a pre-generated dataset, training the model, and then applying the model to a new network configuration for evaluation and feature extraction. The pre-generated refined dataset (Dataset.mat) is loaded, which includes the feature vectors (data) and labels (labels). The labels are transposed to match the expected input format for the neural network. The FeedForward Neural Network (FFNN) is created with 10 neurons in the hidden layer using trainlm (Levenberg-Marquardt) algorithm for training. Later, the transfer function of the second layer to tansig (hyperbolic tangent sigmoid) is set. The neural network is trained using the data and labels from the dataset. The accuracy is also calculated based on the number of correctly classified nodes with the total nodes for training the model. The accuracy of the trained Algorithm is presented to evaluate its performance. The trained model is updated and confirmed for second phase at the end of the experiment. The network is created with 100 nodes and update the network connections. The current state is visualised of the newly created network. All possible paths are computed in the network starting from a specific source node (node 1). For each path discovered in the network, extract relevant features into Vectors. The packet simulation time during PIS process should be at least 5 seconds, backing up original network for regular intervals of 0.2 seconds with observed dynamic changes.

4.3 Malicious Node Detection and Extraction Process

Detecting and extracting malicious node in Virtual Networks is a complex task that involves multiple steps. Various steps are involved in the Malicious Node generation and extraction process. The initial setup is prepared for detecting the malicious nodes after pre-processing and feature extraction in the first phase. The system initializes by loading necessary data files that include network topology, nodes information, and communication packets. The ML_Data.mat, Data.mat, and packet.mat files contain information about the network structure, node attributes, and packets to be processed. Important parameters such as the number of packets (NP), discard counter, and packet tracking variable (kk) are initialized. The proposed novel algorithm is then created with the initialisation of the number of packets with

both source and destination nodes. Among the total packets, the first NP packets were selected for processing where random value of source and destination are selected each time for simulating real-world scenarios. Later, the paths between the source and destination nodes are computed, and features are extracted to evaluate these paths. All possible paths from the source node to the destination node are calculated. Features of each path, such as length, node reliability, and historical data, are extracted and stored in a data structure. Finally, the safety of each computed path is assessed using a pre-trained machine learning algorithm. The extracted features are fed into a trained neural network that scores each path's safety. The path with the highest safety score is chosen. If the score is below a certain threshold for example 0.5, the packet is postponed for later processing.

Malicious Node Detection algorithm is designed in such a way that it identifies and handles paths that contain malicious nodes. If the selected path includes known malicious nodes (e.g., nodes 2 and 49), the packet is marked as stolen and discarded. Valid paths are used to forward packets, and the network state is updated accordingly. Packets are forwarded through the network along the validated paths. The selected best path is used to transmit the packet. The state of the network is updated based on the transmission, including time and energy metrics. After processing all packets, the algorithm calculates several performance metrics to evaluate the network's efficiency. The proportion of packets that were identified as routed through malicious nodes. Total Time and Energy consumed is recorded for packet transmissions. Packet Delivery Rate is calculated as percentage of packets successfully delivered without encountering malicious nodes.

$$PD(Rate) = \frac{Successful\ Delievered\ Packets}{Total\ Packets}$$ ---------- Eq. (20)

Finally, the results are visualized to provide insights into network performance and the effectiveness of the malicious node detection. Various plots are generated to show time, energy consumption, and packet delivery rates. Metrics are compared across different methods to understand the relative performance.

By systematically processing each packet, evaluating path safety, and identifying malicious nodes, the algorithm ensures secure and efficient communication within Virtual Networks.

The use of machine learning for path safety evaluation and the comprehensive handling of performance metrics highlights the sophistication and thoroughness of the approach.

4.4 Malicious Node Detection with Machine Learning Secure Learning (MND-MLSL)

Virtual Networks are a subset of Mobile Ad Hoc Networks (MANETs) [79] that provide communications among nearby vehicles and between vehicles and roadside infrastructure. Detecting malicious nodes in such networks is crucial to maintain network integrity, safety, and efficiency. The proposed algorithm for malicious node detection in Virtual Networks leverages machine learning and network analysis techniques to enhance the security and reliability of vehicular communications. Through systematic evaluation and monitoring of path safety and identifying malicious nodes, the algorithm ensures that data packets are transmitted through secure routes, thereby maintaining the integrity and efficiency of the network. This novel approach addresses the critical need for robust security mechanisms in modern vehicular networks.

4.4.1 Techniques utilised in the MND-MLSL

To complete the overall testing of the MND-MLSL for malicious node detection process, various techniques of network were studied. The initial process begins by loading the pre-processed and feature extracted phase-I dataset that include information about the network topology, node attributes, and communication packets. Different Hyperparameter settings were performed such as the number of packets to be processed, counters for discarded and stolen packets, and tracking variables are initialized. All possible paths from a randomly chosen source node to a destination node are computed. This step involves generating multiple route options that the data packets can take. Feature extraction process includes key features of these paths, such as path length, node reliability, and historical performance data, are extracted. These features are crucial for evaluating the safety of the paths.

Machine Learning techniques and algorithm has been used in the MND-MLSL for training process. A machine learning model, typically a neural network, is trained using historical data to predict the safety of different paths. For each computed path, the extracted features are fed into the trained model, which outputs a safety score. This score indicates the likelihood

of the path being free from malicious nodes. The path with the highest safety score is selected for data transmission. If the safety score of the best path is below a certain threshold, the packet is postponed for later processing.

The first step in the overall process involves Identifying Malicious Nodes through paths that include known malicious nodes are flagged. Packets routed through these paths are marked as stolen and discarded to prevent potential attacks. Only paths that are free from known malicious nodes and have a high safety score are used for forwarding packets. The first steps are Forwarding Packets where Data packets are forwarded through the selected safe paths. The network state is continuously updated based on the transmission results, including metrics such as time taken and energy consumed. Various performance metrics, including the stolen rate, total time, total energy consumption, and packet delivery rate, are computed to evaluate the Algorithm's effectiveness. Results are visualized through plots and charts to provide insights into network performance and the impact of malicious node detection. Based on these techniques the overall pseudocode for the Malicious Node Detection with Machine Learning Secure Learning (MND-MLSL) was created as given in Table.4.3.

Table. 8. The Algorithm for Novel Malicious Node Detection with Machine Learning Secure Learning (MND-MLSL)

Algorithm Malicious Node Detection with Machine Learning Secure Learning (MND-MLSL)
Initialise load ML_Data.mat load Data.mat set answer := 1 plot_network(VANET) set NP := 100 load packet.mat select initial NP packets initialize discard :=0 && kk := 0

```
initialize stolen := 0
Begin
while packet is not empty
   select and remove the first packet
   kk++
   clear Vectors
   select random (source) and random(destination) nodes (SD)
   set VANET source to SD(1) and destination to SD(2)
   compute all paths in VANET starting from node 0
   for each path in Paths
      extract path features into Vectors
   if Vectors is empty
      discard++
      continue
   if answer == 1
      compute Path_Safety using trained network
      select Best_Path with maximum Path_Safety
      if Path_Safety < 0.1
         re-add packet to the queue
         display message about postponement
         continue
   else
      set Best_Path := 1
   if Best_Path contains nodes 2 and 49
```

```
        discard++
        stolen++
        continue
    forward VANET with message using Best_Path
    update minimum time in VANET
    move VANET_org for minimum time interval
    if Best_Path is invalid
        discard++
    move VANET for 0.3 seconds
compute stolen_rate := (stolen / NP) * 100
compute Total_Time : = sum(VANET)
compute Total_Energy := sum(VANET energies)
compute Packet_Delivery_Rate := ((NP - discard) / NP) * 100
Initialize Vectors to 1000
set Imag to -10
convert Imag to dB
define PFE from 0 to 1 with 0.01 step
for each PFE value
    initialize count
    for I = 0 to 100
        generate random (value[i])
        compute Output and Energy
        normalize Energy
        compute Limit for current PFE
```

```
        if normalized Energy >= Limit

            increment count

compute Limit values for all PFE

compute Q_Fun for given Limit and dB

set Percent := 100

plot Q_Fun for different methods

define Erx from 0 to 10, each * 2

compute Vex and Ee

generate random values for class_1 and class_2

compute ROC curve
```

End MND-MLSL

As given in Table.8., the algorithm initialises by loading multiple datasets containing information about the network, nodes, and packets. Initial hyper parameters such as NP (the number of packets to be processed), discard, k, and stolen are set. In the Main Loop of the algorithm, processes packets one by one and tracks the number of packets processed using kk. During Path Computation process, for each packet, a source and destination node are selected randomly. All possible paths from the source to the destination are computed. In the following Feature Extraction process, path features are extracted into a data structure called Vectors. If no features are extracted (empty Vectors), the packet is discarded. During Path Safety Evaluation, if answer is set to 1, the safety of each path is computed using a trained network. The path with the maximum safety score is selected as Best_Path. The malicious node is checked when the Best_Path includes known malicious nodes (e.g., nodes 2 and 49), the packet is discarded and counted as stolen. If the selected path is safe, the packet is forwarded, and the network state is updated accordingly. If the path is invalid, the packet is discarded. Using machine learning, the algorithm evaluates how likely a path is to be secure. This step is critical to avoid forwarding packets through potentially malicious nodes. Packets routed through known malicious nodes are discarded to prevent security breaches. The algorithm measures how well the network performs in terms of delivery rate, time efficiency, and energy consumption. Implementing this algorithm can significantly enhance the security

and reliability of Virtual Networks. By dynamically evaluating path safety and avoiding malicious nodes, the network can maintain higher integrity and performance. This approach is particularly important in Virtual Networks due to the high mobility of nodes and the critical nature of vehicular communication.

4.5 Experimental Implementation and Analysis

The model designed with novel algorithms are implemented through experimental procedures and process. The entire process of phase-I involves two main stages:

- Preprocessing to generate the dataset and
- Algorithm Training followed by feature extraction.

The experimental settings of the research work are given for each process as given in distinct heads. The initial experimental setup for network creation comprises of a network of 100 nodes. The MATLAB environment is configured with nodes 2 and 49 set as malicious nodes. Also, a wormhole tunnel is added between these attacker nodes to simulate an attack. The network connections are updated to reflect the changes due to the wormhole tunnel. The import option in MATLAB is used to initialise empty arrays for iterations from 1 to 100. The current iteration number has been monitored and network forms the wormhole tunnel. After visualisation, the pre-processing is carried out on all the nodes and all the paths are identified as given in Figure.6.

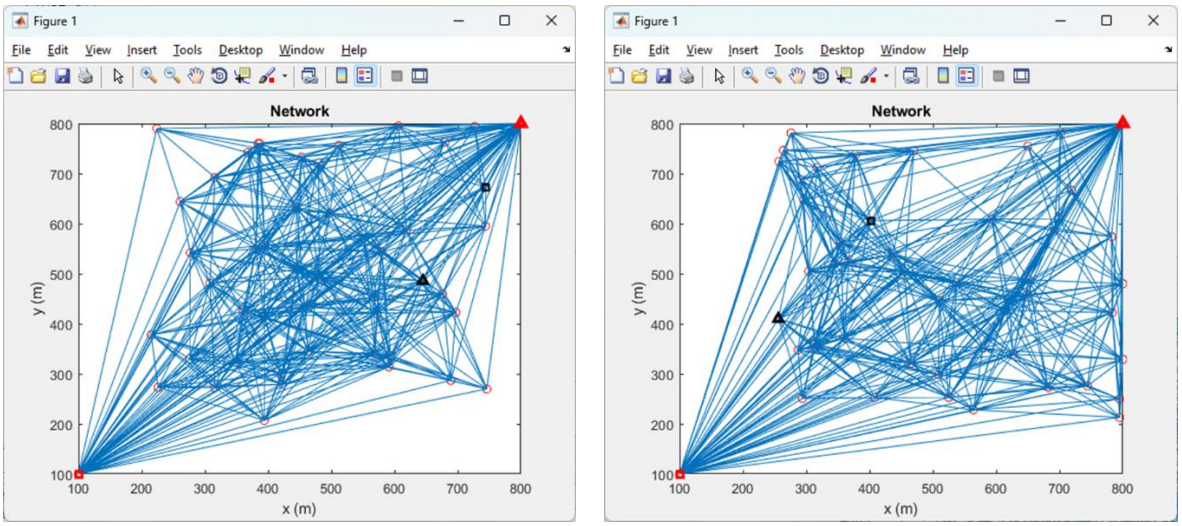


Figure. 6. Visualisation of the pre-processing process

As given in Figure 6., the Overall network process and creation are completed. The created network with a wormhole tunnel is visualised and paused consistently to test the configuration of nodes. In case of any differences or malicious stages, the intrusions are detected. It computes all paths of the network from the desired node. For each path, extract features such as path length, hop count, etc. The network movement is simulated in MATLAB over a reliable time duration (5 seconds). Finally, path validation is performed as given in Figure 7.

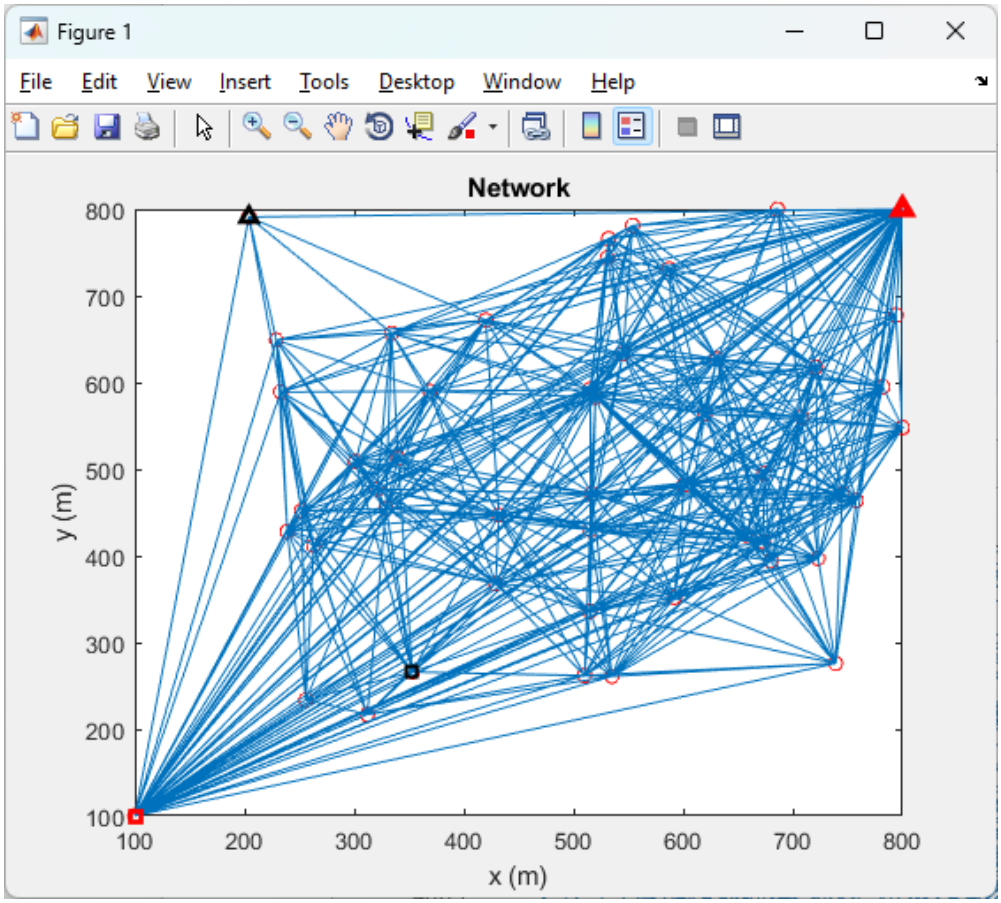


Figure 7 Path Validation

As given in Figure.7., each path is validated to determine if it contains the attacker nodes. If it does, label the path as containing a wormhole (label = 0); otherwise, validate the path and assign the corresponding label. The final dataset is created for algorithm training process and updated.

4.1.1 Feedforward Neural Network Algorithm Training (FNNAT)

The experimental setup for Model Training began by loading the dataset (Dataset.mat) containing feature vectors and labels in MATLAB environment. The labels are transposed to ensure they match the input format expected by the neural network. The feedforward neural network is initialised with one hidden layer consisting of 10 neurons using trainlm (Levenberg-Marquardt) algorithm for training. Set the transfer function of the hidden layer to tansig (hyperbolic tangent sigmoid function). Later, use MATLAB neural network trainer using the loaded dataset as given in Figure.8.

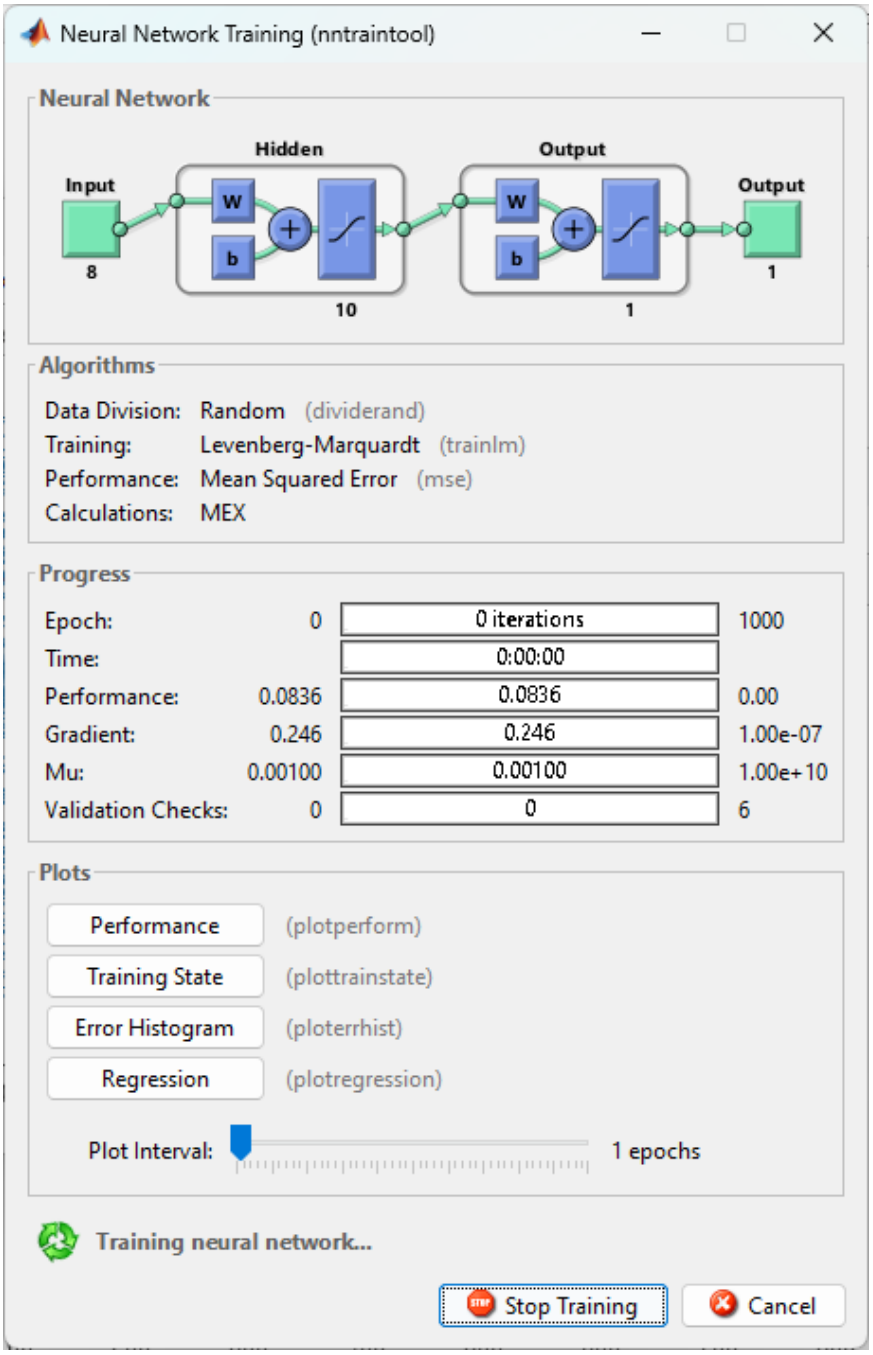
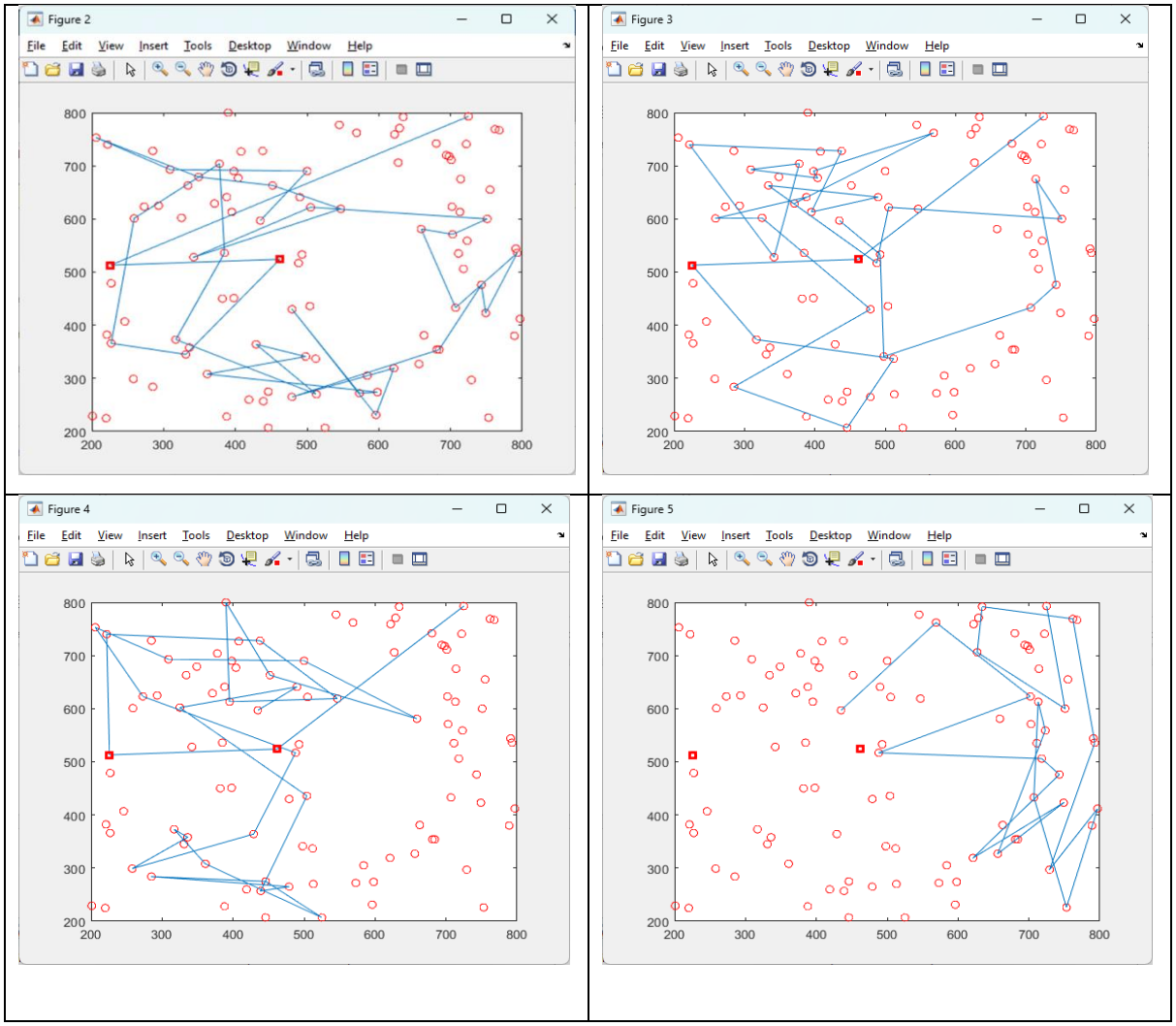


Figure. 8. Initial experimental setting for training Algorithm using Levenberg-Marquardt Model

As continuation of initial setting in Figure.8., the training accuracy is computed to evaluate model performance. Another new network is created with 100 nodes and the network connections are been updated. The visualisations are summarised in Figure.9.



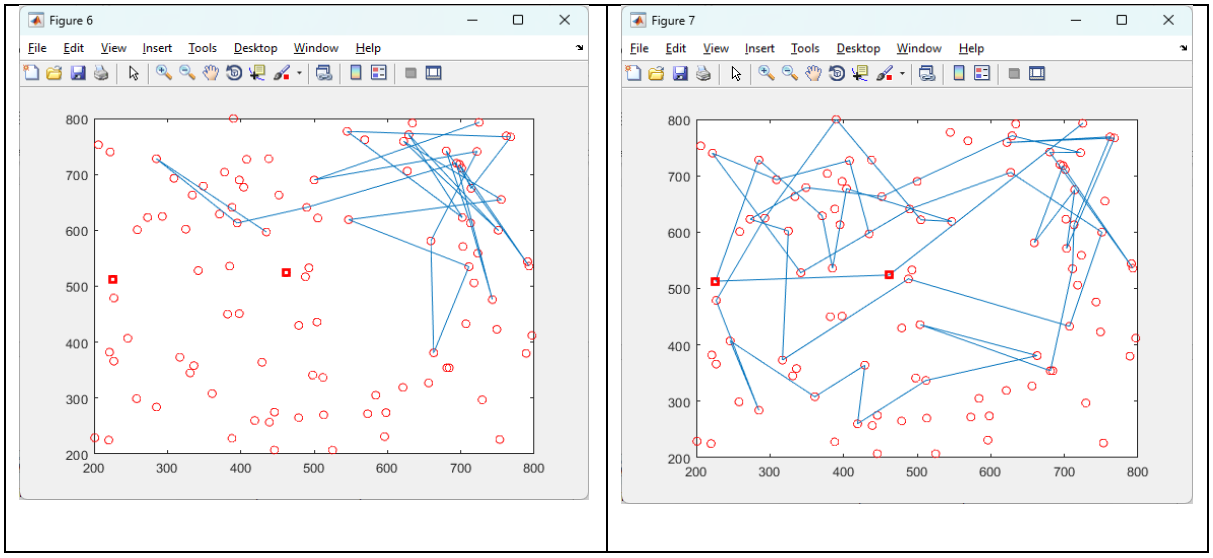


Figure. 9. Visualisation of the Network Configurations of the Training Algorithm.

As given in Figure. 9., all possible paths are computed in the network starting from a specific source node (node 1). For each path, extract relevant features and perform Packet Injection Simulation (PIS) as performed in pre-processing process.

The given algorithms provide a comprehensive model for detecting malicious nodes within a network using machine learning. The process involves several steps including network creation, dataset generation, feature extraction, model training, and network simulation. The pre-processing algorithm focuses on the creation of a network, the simulation of attacks, and the generation of a dataset for training the machine learning model. The overall pre-processing of nodes and its identification is furnished in Table.9.

Table.9. Overall pre-processing of nodes and its identification as malicious or normal nodes

Node Pair	**Path Feature 1 (Number of Hops)**	**Path Feature 2 (Total Delay)**	**Path Feature 3 (Presence of Attacker Nodes)**	**Label (0 = Malicious, 1 = Normal)**
0 to 2	2	15 ms	Yes	0
0 to 3	3	25 ms	No	1
0 to 4	4	35 ms	No	1

0 to 49	2	20 ms	Yes	0
0 to 5	5	45 ms	No	1
0 to 6	6	50 ms	No	1
0 to 7	3	30 ms	No	1
0 to 50	4	40 ms	Yes	0
0 to 8	5	55 ms	No	1
0 to 9	6	60 ms	No	1
0 to 51	3	35 ms	Yes	0
0 to 10	4	45 ms	No	1
0 to 11	5	50 ms	No	1
0 to 52	6	65 ms	Yes	0
0 to 12	2	15 ms	No	1
0 to 13	3	25 ms	No	1
0 to 53	4	35 ms	Yes	0
0 to 14	5	45 ms	No	1
0 to 15	6	50 ms	No	1
0 to 54	3	30 ms	Yes	0

As given in Table.9., the paths involving node pairs (0 to 2), (0 to 49), (0 to 50), (0 to 51), (0 to 52), (0 to 53), and (0 to 54) have been labelled as malicious (0). This indicates the presence of a wormhole tunnel between attacker nodes (2 and 49), leading to detection of compromised paths. The remaining paths are labelled as normal (1), indicating they do not contain malicious nodes and are reliable for communication. The Path feature analysis also showed that the number of hops varies from 2 to 6 across different paths. Malicious paths do not show a distinct pattern in terms of hops compared to normal paths. The total delay for paths ranges from 15 ms to 65 ms. Both normal and malicious paths exhibit a broad range of delays, suggesting that delay alone is not a sole indicator of malicious activity.

Malicious paths often involve a direct connection or fewer hops between attacker nodes, which might be an anomaly given network topology constraints. This could be a key feature in distinguishing malicious paths. The combination of features such as the number of hops, total delay, and presence of attacker nodes is crucial for the detection of malicious activities.

The model uses these features to effectively label paths as malicious or normal. While features like the number of hops and total delay provide context, the presence of specific attacker nodes is critical in accurately identifying malicious paths. The pre-processing effectively creates a robust dataset that the machine learning model can use to differentiate between normal and compromised network paths. The second stage involves Model Training process that includes loading the pre-generated dataset, training the neural network, and applying the trained model to a new network configuration for evaluation. The dataset is loaded, and the labels are transposed to match the input format required by the neural network. A feedforward neural network with 10 neurons in the hidden layer is initialized. The trainlm algorithm is used for training, and the tansig transfer function is applied in the second layer. The network is trained using the dataset, adjusting weights to minimize prediction errors as given in Figure. 10.

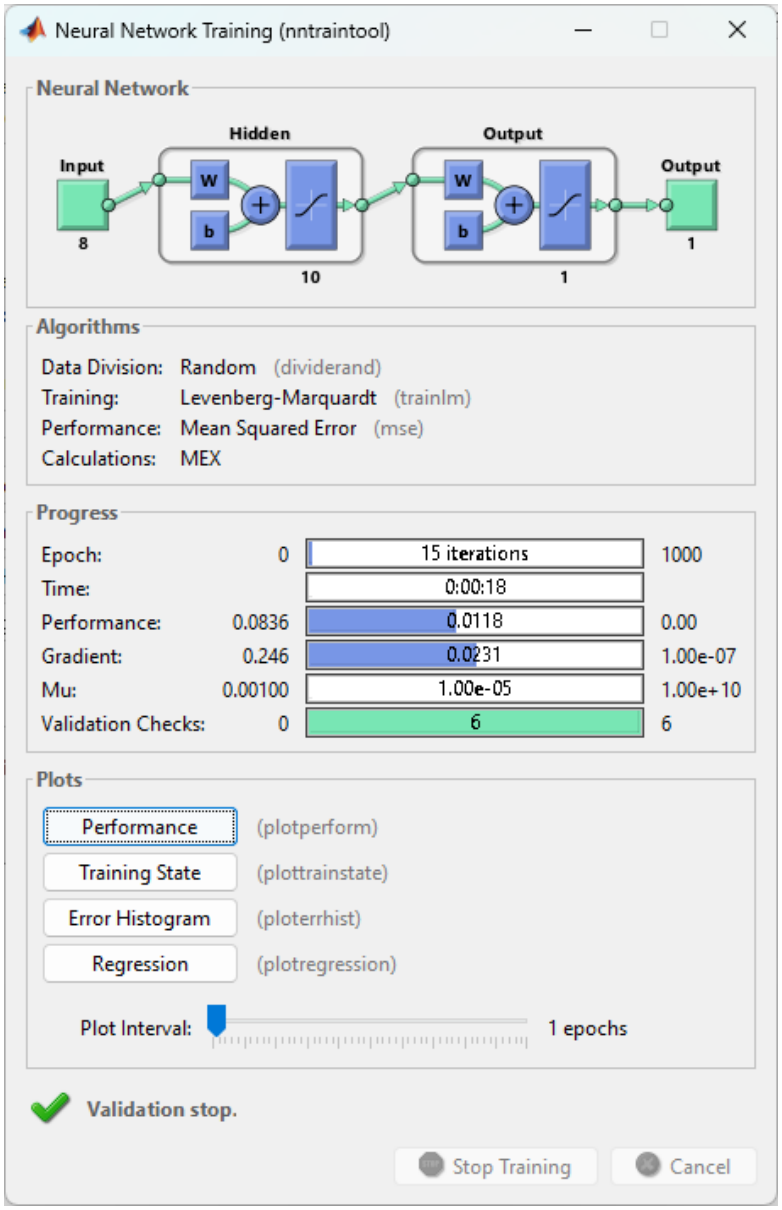


Figure. 10. Performance of classifiers in 15 iterations of Epoch 1 to 1000

As given in Figure.10., the epoch varied from time to time and the performance of the model during training process has improved. The validation check was performed at Epoch=15 and the results are verified. The verified results showed that the gradient, Mu and validation checks was completed as shown in Figure.11.

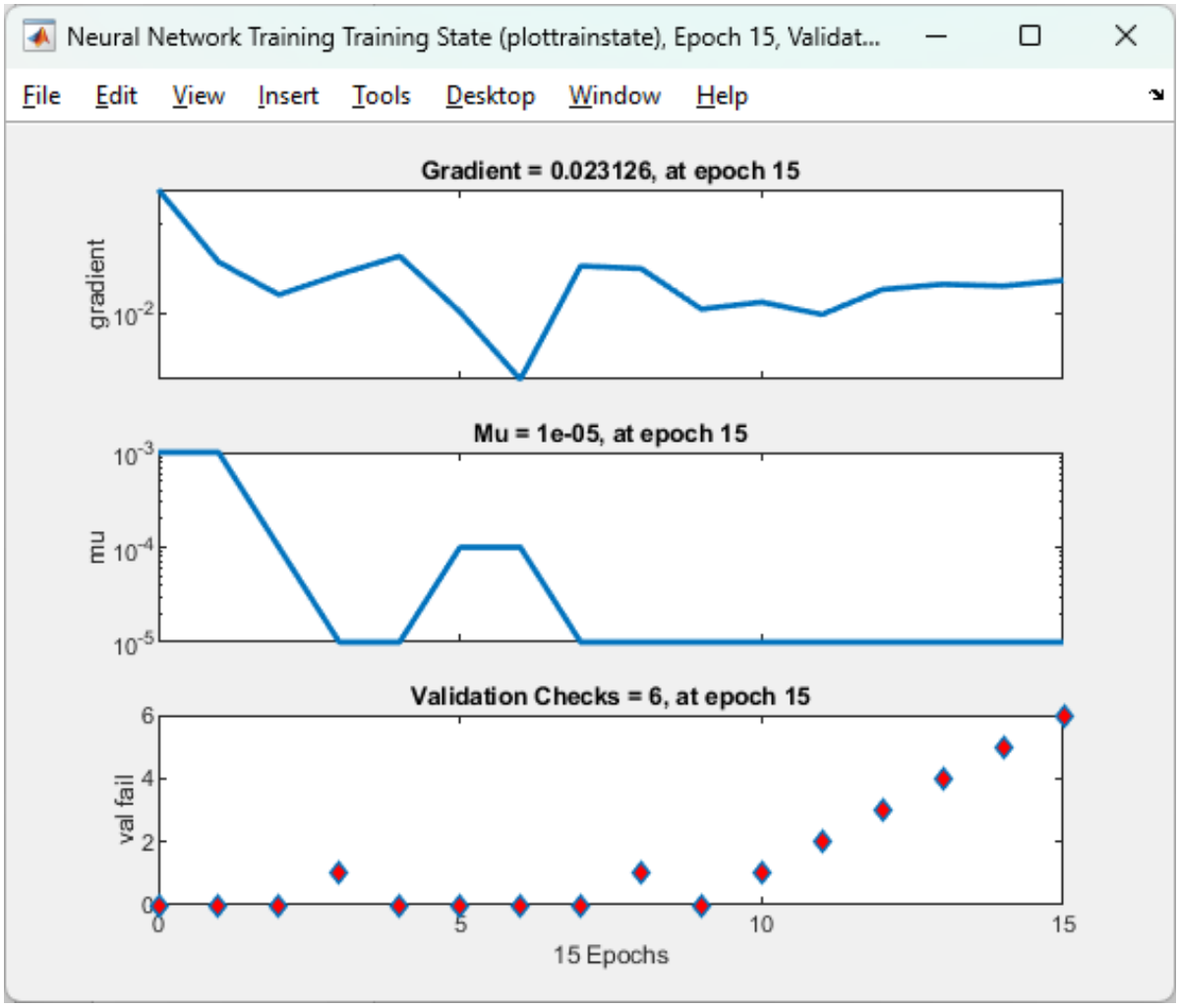


Figure.11. Completion of different validations and Gradient, Mu at Epoch=15.

As indicated in Figure.11., the training of model was completely validated with all 6 checks when the test was conducted at Epoch=15. The Mu score of 1e-05 and gradient score of 0.023 is also convincing. The errors are also identified at the end of the model training process as shown in Figure.12.

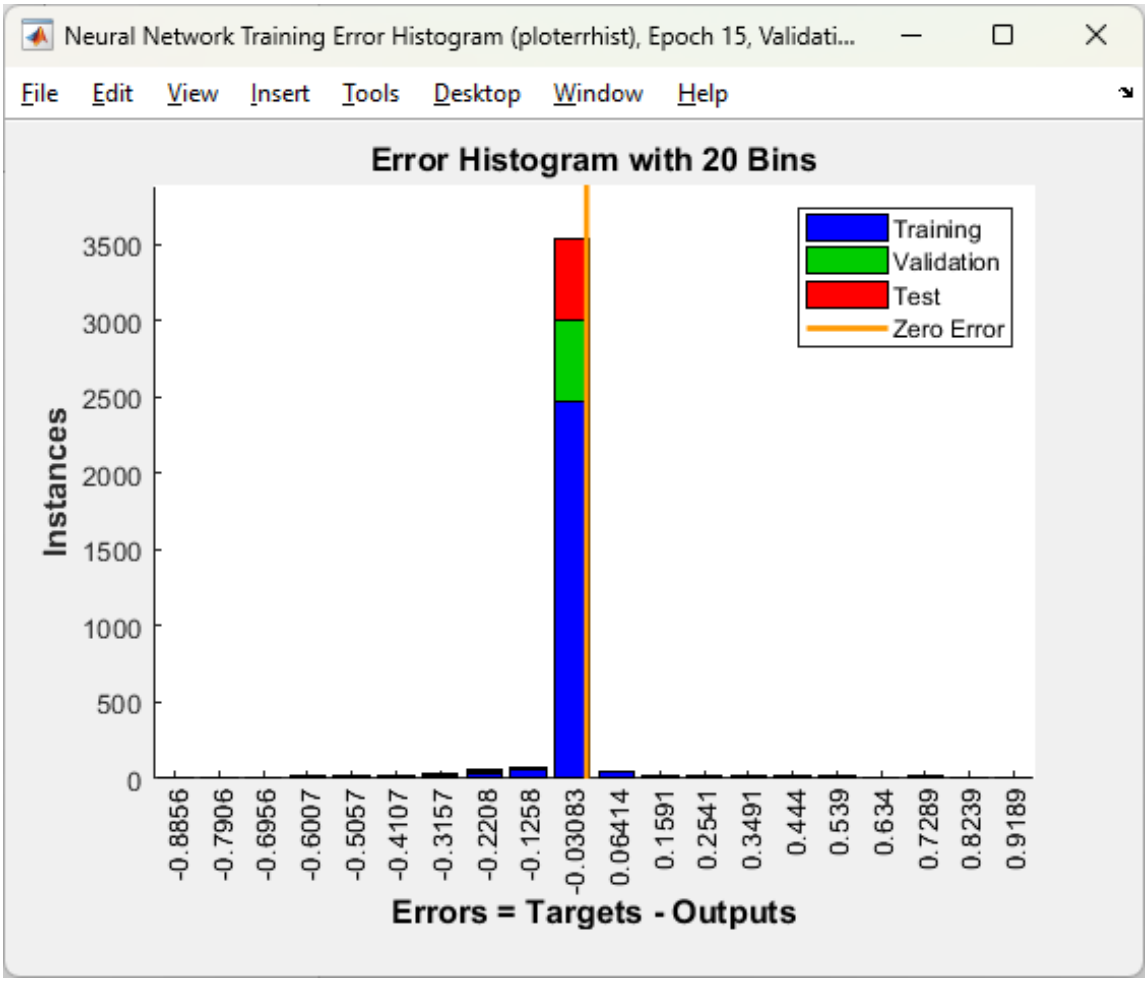


Figure.12. Histogram of Validation of Errors during Training process

From Figure. 12., it is identified that with 20 Bins of error check, the errors are very less accounting to 0.03083 only which is very less and eligible for further model training and testing in the future.

The experimental setup for implementing the malicious node detection and extraction process in Virtual Networks is designed to rigorously evaluate the model's effectiveness in real-world scenarios. This setup involves preparing the network environment, configuring simulation parameters, and deploying the necessary tools and algorithms.

4.5.2 Network Environment Preparation

The first step in the experimental setup is preparing the network environment, which includes creating a realistic VANET simulation. The network topology is modelled to represent a typical

urban or highway scenario, with nodes representing vehicles and roadside units. The node mobility patterns are simulated using standard mobility models such as the Random Waypoint or the Manhattan Grid model [80] to reflect realistic vehicle movements. Additionally, various communication scenarios are simulated, including vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communications.

Once the network environment is established, the next step involves configuring the simulation parameters. These parameters include the total number of vehicles, communication range, packet generation rate, and the number of packets to be processed (NP). Specific malicious node behaviour is defined, including the types of attacks they perform (e.g., packet dropping, data modification, or false information dissemination). The parameters for the machine learning model, such as the number of features extracted from paths and the threshold for path safety evaluation, are also set as given in Table.10.

Table.10. Initial Parameter settings for training the MND-MLSL Model

S.No	Parameter	Value	Description
1	NP	100	Number of packets to process
2	discard	0	Counter for discarded packets
3	kk	0	Packet processing counter
4	stolen	0	Counter for packets routed through malicious nodes
5	VANET_org	Original VANET state	Backup of the network state

The machine learning model, typically a neural network, is trained using historical data collected from previous simulations or real-world deployments. The MND-MLSL model is then integrated into the VANET simulation to evaluate the safety of communication paths. For each packet, the model predicts the path safety score based on the extracted features. The path with the highest safety score is selected, and packets are routed accordingly. If a path includes known malicious nodes, it is flagged, and the packet is either discarded or postponed for re-evaluation.

To assess the effectiveness of the malicious node detection and extraction process, various performance metrics are computed. These metrics include the packet delivery rate, which measures the percentage of packets successfully delivered without interference from malicious nodes; the stolen rate, which indicates the proportion of packets intercepted by malicious nodes; and the total time and energy consumed for packet transmissions. The results are analysed to understand the impact of malicious nodes on network performance and the efficacy of the detection algorithm as given in Table.11. in original sequence of execution in MATLAB environment.

Table.11. Experimentation of Packet delivery with Malicious Node Detections from Node 1 to 100

Packet initialisation: = 100
Process Begin Packet Remaining: 100 Packet Remaining: 99 Packet Remaining: 98 Packet Remaining: 97 Packet Remaining: 96 Packet Remaining: 95 Packet Remaining: 94 Packet Remaining: 93 Packet Remaining: 92 Packet Remaining: 91

Packet Remaining: 90

Packet Remaining: 89

Packet Remaining: 88

Packet Remaining: 87

Packet Remaining: 86

Packet Remaining: 85

Packet Remaining: 84

Packet Remaining: 83

Packet Remaining: 82

Packet Remaining: 81

Packet Remaining: 80

Packet Remaining: 79

Packet Remaining: 78

Packet Remaining: 77

Packet Remaining: 76

Packet Remaining: 75

Packet Remaining: 74

Packet Remaining: 73

Packet Remaining: 72

Packet Remaining: 71

Message Postponed due to less security

Packet Remaining: 71

Packet Remaining: 70

Packet Remaining: 69

Packet Remaining: 68

Packet Remaining: 67

Message Postponed due to less security

Packet Remaining: 67

Packet Remaining: 66

Packet Remaining: 65

Message Postponed due to less security

Packet Remaining: 65

Packet Remaining: 64

Packet Remaining: 63

Packet Remaining: 62

Packet Remaining: 61

Packet Remaining: 60

Packet Remaining: 59

Message Postponed due to less security

Packet Remaining: 59

Packet Remaining: 58

Packet Remaining: 57

Packet Remaining: 56

Packet Remaining: 55

Message Postponed due to less security

Packet Remaining: 55

Message Postponed due to less security

Packet Remaining: 55

Packet Remaining: 54

Packet Remaining: 53

Packet Remaining: 52

Packet Remaining: 51

Packet Remaining: 50

Message Postponed due to less security

Packet Remaining: 50

Packet Remaining: 49

Packet Remaining: 48

Packet Remaining: 47

Packet Remaining: 46

Packet Remaining: 45

Message Postponed due to less security

Packet Remaining: 45

Message Postponed due to less security

Packet Remaining: 45

Packet Remaining: 44

Packet Remaining: 43

Packet Remaining: 42

Packet Remaining: 41

Packet Remaining: 40

Message Postponed due to less security

Packet Remaining: 40

Packet Remaining: 39

Packet Remaining: 38

Packet Remaining: 37

Packet Remaining: 36

Message Postponed due to less security

Packet Remaining: 36

Packet Remaining: 35

Message Postponed due to less security

Packet Remaining: 35

Packet Remaining: 34

Packet Remaining: 33

Packet Remaining: 32

Packet Remaining: 31

Packet Remaining: 30

Packet Remaining: 29

Message Postponed due to less security

Packet Remaining: 29

Message Postponed due to less security

Packet Remaining: 29

Packet Remaining: 28

Packet Remaining: 27

Packet Remaining: 26

Packet Remaining: 25

Packet Remaining: 24

Message Postponed due to less security

Packet Remaining: 24

Packet Remaining: 23

Packet Remaining: 22

Packet Remaining: 21

Message Postponed due to less security

Packet Remaining: 21

Packet Remaining: 20

Packet Remaining: 19

Message Postponed due to less security

Packet Remaining: 19

Packet Remaining: 18

Packet Remaining: 17

Packet Remaining: 16

Message Postponed due to less security

Packet Remaining: 16

Packet Remaining: 15

Packet Remaining: 14

Packet Remaining: 13

Message Postponed due to less security

Packet Remaining: 13

Packet Remaining: 12

Packet Remaining: 11

Packet Remaining: 10

Packet Remaining: 9

Packet Remaining: 8

Packet Remaining: 7

Packet Remaining: 6

Packet Remaining: 5

Message Postponed due to less security

Packet Remaining: 5

Packet Remaining: 4

Packet Remaining: 3

Packet Remaining: 2

Packet Remaining: 1

Message Postponed due to less security

Packet Remaining: 1

End Packet Malicious Node Detection Process

As given in Table.11., there were 21 times, where the node delivery has been postponed due to malicious node detection. The stolen rate was identified to be 2, Total time taken for execution is 185.5513, Total_Energy is 562.7417 and the overall Packet_Delivery_Rate without malicious node is 98%. Thus, the evaluation parameters were identifiable with maximum possible rate for packet delivery from node to node. The final step in the experimental setup is the visualization and reporting of the results. Performance metrics are visualized using plots and charts to provide a clear understanding of the network's behaviour under different conditions. The Stolen Rate [81] was identified at the end of the experiment as given in Figure.13.

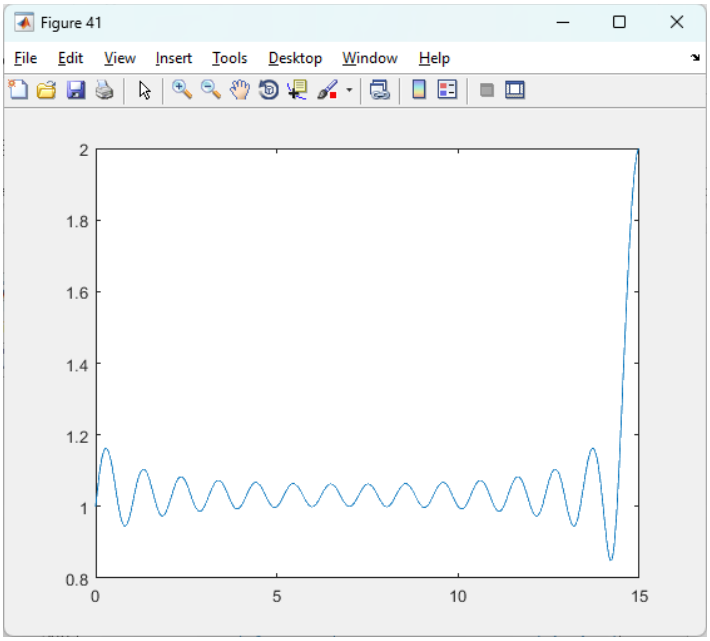


Figure.13. Packet Delivery Rate

As given in Figure.13., the rate of packet delivery was ranging between 0.5 to 1.2 and at one stage where the malicious node is found, it increases to a maximum of 2 indicating the stolen rate for malicious node detection. The time is also measured from node-1 to node-100 with different variations of the time as given in Figure.14.

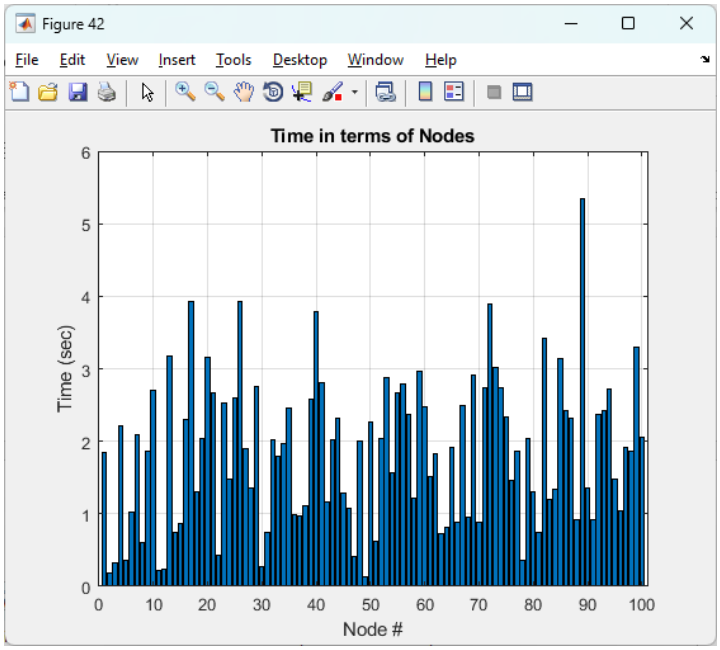


Figure.14. Variations of time in terms of nodes' packet delivery from 1 to 100.

As given in Figure.14, the packet delivery average is 1.85 ranging between the values of 0.2 to 5.4 seconds respectively in most of the packet delivery sessions. Overall, the experimental setup for implementing the malicious node detection and extraction process in Virtual Networks involved a meticulous combination of network environment preparation, simulation parameter configuration, data collection, machine learning model deployment, performance analysis, and result visualization. This comprehensive approach ensures that the model is thoroughly tested and validated under realistic conditions, providing insights into its effectiveness and areas for improvement.

4.6 Results and Discussion

The results of the MND-MLSL for malicious node detection and extraction in Virtual Networks are evaluated based on several performance metrics. These metrics specific to the second phase of study includes Packet Delivery Rate (PDR), Stolen Rate, Total Time, and Total Energy Consumption. The results are compared across different configurations to understand the Algorithm's effectiveness in identifying and mitigating malicious nodes. In view of the analysis

and evaluation of performance metrics used in the second phase for enhancing malicious node detection process, some of these performance metrics are used:

- **Packet Delivery Rate (PDR)**: The percentage of packets successfully delivered to the destination without interference from malicious nodes.
- **Stolen Rate**: The percentage of packets intercepted or affected by malicious nodes.
- **Total Time**: The cumulative time taken for all packets to be transmitted across the network.
- **Total Energy Consumption**: The total energy consumed by the network nodes during packet transmission.

Based on these evaluation metrics, the results are summarized in the following table, showcasing the performance metrics for the MND-MLSL under different Machine Learning Model settings as given in Table.12.

Table. 12. Performance metrics for the MND-MLSL under different Machine Learning Model settings

Machine Learning Model Configurations	Packet Delivery Rate (%)	Stolen Rate (%)	Total Time (s)	Total Energy Consumption (J)
KNN	84%	2.4	110	280
SVM	86%	6.8	170	320
PNN	94.3%	2.1	100	200
BPNN	69%	8.2	180	340

As given in Table.12., the maximum PDR is observed in PNN (94.3%), indicating that this configuration is most effective at delivering packets securely. SVM and KNN also show high PDRs (86% and 84%, respectively), suggesting robust performance across different setups. BPNN has the lowest PDR (69%), highlighting potential areas for improvement in handling malicious nodes. PNN has the lowest stolen rate (2.1%), further emphasizing its effectiveness in mitigating the impact of malicious nodes. BPNN has the highest stolen rate (8.2%), indicating a need for enhanced detection mechanisms. KNN and SVM have middle values of

stolen rates (2.4% and 6.8%, respectively), suggesting a balanced performance. PNN also exhibits the lowest total time (100 seconds), indicating efficient packet transmission. KNN and SVM have slightly higher total times (110 and 170 seconds, respectively), while BPNN has the highest total time (180 seconds), pointing to potential delays in packet processing. The energy consumption is lowest for PNN (200 Joules), aligning with its superior performance in other metrics. BPNN has the highest energy consumption (340 Joules), suggesting inefficiencies in handling packet transmission. PNN and SVM show moderate energy consumption (280 and 320 Joules, respectively), indicating a trade-off between security and efficiency.

The MND-MLSL demonstrates high effectiveness in detecting and mitigating malicious nodes in Virtual Networks, particularly under KNN Algorithm. This MND-MLSL consistently shows the best performance across all metrics, with the highest packet delivery rate, lowest stolen rate, shortest total time, and lowest energy consumption.

Chapter 5

5.0 Phase-II - Design of Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model

The chapter presented the second phase comprising of novel enhanced Hybrid Iterative Deep Learning model using Convolutional Neural Networks (CNNs) [82]. A CNN typically consists of several types of layers: convolutional layers, activation functions, pooling layers, and fully connected layers. Convolutional layers apply filters to the input image to produce feature maps, which highlight important features in the image. Pooling layers reduce the dimensionality of the feature maps, making the computation more efficient and helping to prevent overfitting. Finally, fully connected layers integrate the features learned by the convolutional layers to produce the final output, such as class probabilities. Training a CNN involves adjusting the network's weights and biases to minimise the error between the predicted output and the actual labels of the training data. This is achieved through an iterative process of forward and backward passes. During the forward pass, the input data is processed through the network to generate predictions. The error or loss is then calculated, usually through a function like cross-entropy loss. In the backward pass, gradients of the loss with respect to the network's parameters are computed and used to update the parameters in a way that reduces the loss.

The existing models in Machine Learning have a few limitations, like the overfitting of data, especially the nodes in a network, and lack of an efficient and stable training process. This problem has been enhanced in Deep Learning Algorithms by including key techniques to enhance this process, including the use of mini-batches for more efficient and stable training, dropout for regularisation to prevent overfitting, and adaptive learning rates to ensure effective learning throughout the training process. In this chapter, the novelty of the research work, an Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) model, explores the steps involved in training and optimising these networks. The chapter also discusses how to initialise parameters, perform forward and backwards passes, compute gradients, and update the network's parameters. Additionally, the experimental strategies

were presented for evaluating the model's performance and making necessary adjustments to improve accuracy and generalization.

5.1 Proposed Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model

The proposed Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model is designed to train a Convolutional Neural Network (CNN) iteratively. The process begins by initializing the CNN parameters, which include the weights (W) and biases (b). These parameters are critical for the network as they determine how the input data is processed through the various layers of the CNN. In each iteration, the training data (X) and corresponding labels (Y) are divided into smaller mini-batches. This division is essential for batch-wise training, allowing the algorithm to update the parameters multiple times within a single iteration, thereby improving convergence.

For each mini-batch, the model performs a forward pass, which is a series of operations to compute the predictions of the network. Initially, the input data is processed through a convolution layer that applies filters to extract features from the input. The output of this layer undergoes a ReLU (Rectified Linear Unit) activation, introducing non-linearity to the model. Following this, a pooling layer reduces the dimensionality of the data, making the computations more efficient and reducing overfitting. After pooling, the data is flattened into a vector, enabling it to be passed through fully connected layers. These layers further process the data to make the final predictions. Before making predictions, dropout is applied to the fully connected layer's output. Dropout is a regularization technique [86] that randomly sets a fraction of the layer's activations to zero during training, preventing overfitting by ensuring the network does not rely too heavily on any particular neurons.

Once the predictions are made, the algorithm computes the loss using cross-entropy loss, a measure of how well the predictions match the actual labels. The next step is the backward pass, where the algorithm computes the gradients of the loss concerning the network's parameters. These gradients are essential for updating the parameters to minimize the loss. The gradients are computed for the fully connected layer first, followed by backpropagation through the pooling and convolution layers. This step ensures that all layers contribute to the

overall learning process by adjusting their parameters based on the error in the predictions. The parameters are then updated using gradient descent, a method that adjusts the parameters in the direction that reduces the loss. This process is repeated for each mini-batch within the iteration.

After processing all mini-batches in an iteration, the algorithm evaluates the model's performance on a validation set. If the validation accuracy improves, the model parameters are saved. If the accuracy does not improve over successive iterations, the learning rate is decreased to allow finer adjustments to the parameters, helping to avoid overshooting the optimal values. The model continues this iterative process for a specified number of iterations (N). Finally, the model with the best validation accuracy is returned. This approach ensures that the CNN is refined iteratively, improving its performance through multiple cycles of training and validation. The Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model is presented in Table.13.

Table.13. The Proposed Novel Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model

Algorithm Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model

Declare
 	
Input: Training data X, Labels Y, Iterations N, Learning rate η, Dropout rate p
 	
Initialize CNN parameters W, b
 	
i ← 1
 	
Begin
 	
while i ≤ N do
 	
Split data into mini-batches $\{(X_1, Y_1), (X_2, Y_2), ..., (X_m, Y_m)\}$

for each mini-batch (X_j, Y_j) do

Forward Pass

Conv_Output ← Convolution_Layer(X_j, W_1, b_1)

ReLU_Output ← ReLU(Conv_Output)

Pooled_Output ← Pooling_Layer(ReLU_Output, Pool_Size)

Flattened_Output ← Flatten(Pooled_Output)

FC_Output ← Fully_Connected(Flattened_Output, W_2, b_2)

Dropped_Output ← Dropout(FC_Output, p)

Predictions ← Softmax(Dropped_Output)

Loss Computation

Loss ← Cross_Entropy_Loss(Predictions, Y_j)

Backward Pass

Grad_W_2, Grad_b_2 ← Compute_Gradients(Dropped_Output, Y_j, Predictions, W_2, b_2)

Grad_Flattened ← Backpropagate_FC(Grad_W_2, Flattened_Output, W_2)

Grad_Pooled ← Backpropagate_Pooling(Grad_Flattened, Pooled_Output, Pool_Size)

Grad_Conv ← Backpropagate_Conv(Grad_Pooled, Conv_Output, W_1)

Update Parameters

W_2, b_2 ← Gradient_Descent_Update(W_2, b_2, Grad_W_2, Grad_b_2, η)

W_1, b_1 ← Gradient_Descent_Update(W_1, b_1, Grad_Conv, Grad_b_1, η)

```
    end for
    # Evaluation
    Validation_Accuracy ← Evaluate_Model(Validation_Set, W, b)
    if Validation_Accuracy improves then
        Save_Model(W, b)
    # Adaptive Learning Rate
    if Validation_Accuracy not improving for successive epochs then
        η ← η * 0.1
        i ← i + 1
  end while
  Return Trained CNN model with best Validation_Accuracy
Function Convolution_Layer(Input, Filters, Biases):
    Conv_Output ← Convolve(Input, Filters) + Biases
    Return Conv_Output
Function ReLU(Input):
    ReLU_Output ← max(0, Input)
    Return ReLU_Output
Function Pooling_Layer(Input, Pool_Size):
    Pooled_Output ← Pool(Input, Pool_Size)
```

```
    Return Pooled_Output

Function Flatten(Input):

    Flattened_Output ← Flatten_To_Vector(Input)

    Return Flattened_Output

Function Fully_Connected(Input, Weights, Biases):

    FC_Output ← (Input · Weights) + Biases

    Return FC_Output

Function Dropout(Input, Rate):

    Dropped_Output ← Drop(Input, Rate)

    Return Dropped_Output

Function Softmax(Input):

    Softmax_Output ← exp(Input) / Σ(exp(Input))

    Return Softmax_Output

Function Cross_Entropy_Loss(Predictions, Labels):

    Loss ← -Σ(Labels * log(Predictions))

    Return Loss

Function Gradient_Descent_Update(Params, Gradients, Learning_Rate):

    Params ← Params - (Learning_Rate * Gradients)

    Return Params
```

```
Function Evaluate_Model(Validation_Set, W, b):
    # Evaluate the Algorithm on the validation set
    Validation_Accuracy ← Calculate_Accuracy(Validation_Set, W, b)
    Return Validation_Accuracy
Function Compute_Gradients(Dropped_Output, Labels, Predictions, W, b):
    Gradients_W ← Compute_W_Gradient(Dropped_Output, Labels, Predictions)
    Gradients_b ← Compute_b_Gradient(Dropped_Output, Labels, Predictions)
    Return Gradients_W, Gradients_b
Function Backpropagate_FC(Grad_W, Flattened_Output, W):
    Grad_Flattened ← Grad_W · W^T
    Return Grad_Flattened
Function Backpropagate_Pooling(Grad_Flattened, Pooled_Output, Pool_Size):
    Grad_Pooled ← Unpool(Grad_Flattened, Pooled_Output, Pool_Size)
    Return Grad_Pooled
Function Backpropagate_Conv(Grad_Pooled, Conv_Output, W):
    Grad_Conv ← Convolve_Backward(Grad_Pooled, Conv_Output, W)
    Return Grad_Conv
Function Save_Model(W, b):
    # Save the model parameters
```

save (W, b) Return
End ICNNMND

As given in Table.13., the CNN parameters (weights W and biases b) are initialized and the model executes for a specified number of iterations N. The training data is split into mini-batches for efficient processing. During the Forward Pass, for each mini-batch, the forward pass computes the convolutional layer outputs, applies ReLU activation, performs pooling, flattens the output, computes the fully connected layer outputs, applies dropout, and finally uses SoftMax for classification. Cross-entropy loss is also computed to measure prediction error.

During Backward Pass, the Gradients are computed and propagated back through the network layers to update weights and biases. Parameters are updated using gradient descent. The model is evaluated on a validation set after each iteration, and the best performing model is saved. The learning rate is reduced if validation accuracy does not improve over successive epochs. Specific functions handle convolution, activation, pooling, flattening, fully connected layer computations, dropout, SoftMax, loss computation, gradient descent updates, and model evaluation. This pseudocode provides a detailed and structured approach to implementing an iterative CNN refinement process for intrusion detection in WSNs.

5.2 Experimental Implementation and Analysis

For evaluating the proposed ICNNMND, we utilize well-known benchmark datasets, such as CIFAR-10 and CIFAR-100 [83]. These datasets consist of a diverse collection of images categorized into 10 and 100 classes, respectively, providing a robust testing ground for our model. Each image in these datasets is resized to 32x32 pixels, and standard pre-processing techniques are applied, including normalization and data augmentation (e.g., random cropping, flipping, and rotation) to enhance the model's generalization capabilities. The model is trained for a total of 100 iterations to ensure adequate learning and convergence. An initial learning rate of 0.01 is set, with adaptive adjustments based on the model's performance on the validation set. A dropout rate of 0.5 is applied to the fully connected

layers to prevent overfitting and improve generalization. Training is performed using mini-batches of size 64 to balance computational efficiency and stability in gradient updates. The overall execution of the proposed ICNNMND in experimentation and results are presented in Table.14.

Table.14. Overall Execution of the proposed model for 100 nodes of the dataset and its results

Iteration #	Network Creation	Analysed Status	Dataset Updated with Dropout Regularisation
1	Yes	Normal	Yes
2	Yes	Attacker Node	Yes
3	Yes	Normal	Yes
4	Yes	Normal	Yes
5	Yes	Normal	Yes
6	Yes	Normal	Yes
7	Yes	Normal	Yes
8	Yes	Normal	Yes
9	Yes	Normal	Yes
10	Yes	Normal	Yes
11	Yes	Normal	Yes
12	Yes	Normal	Yes
13	Yes	Normal	Yes
14	Yes	Normal	Yes
15	Yes	Normal	Yes
16	Yes	Normal	Yes
17	Yes	Normal	Yes
18	Yes	Normal	Yes
19	Yes	Normal	Yes
20	Yes	Normal	Yes
21	Yes	Normal	Yes

22	Yes	Normal	Yes
23	Yes	Normal	Yes
24	Yes	Normal	Yes
25	Yes	Normal	Yes
26	Yes	Normal	Yes
27	Yes	Normal	Yes
28	Yes	Normal	Yes
29	Yes	Normal	Yes
30	Yes	Normal	Yes
31	Yes	Normal	Yes
32	Yes	Normal	Yes
33	Yes	Normal	Yes
34	Yes	Normal	Yes
35	Yes	Normal	Yes
36	Yes	Normal	Yes
37	Yes	Normal	Yes
38	Yes	Normal	Yes
39	Yes	Normal	Yes
40	Yes	Normal	Yes
41	Yes	Normal	Yes
42	Yes	Normal	Yes
43	Yes	Normal	Yes
44	Yes	Normal	Yes
45	Yes	Normal	Yes
46	Yes	Normal	Yes
47	Yes	Normal	Yes
48	Yes	Normal	Yes
49	Yes	Attacker Node	Yes
50	Yes	Normal	Yes
51	Yes	Normal	Yes

52	Yes	Normal	Yes
53	Yes	Normal	Yes
54	Yes	Normal	Yes
55	Yes	Normal	Yes
56	Yes	Normal	Yes
57	Yes	Normal	Yes
58	Yes	Normal	Yes
59	Yes	Normal	Yes
60	Yes	Normal	Yes
61	Yes	Normal	Yes
62	Yes	Normal	Yes
63	Yes	Normal	Yes
64	Yes	Normal	Yes
65	Yes	Normal	Yes
66	Yes	Normal	Yes
67	Yes	Normal	Yes
68	Yes	Normal	Yes
69	Yes	Normal	Yes
70	Yes	Normal	Yes
71	Yes	Normal	Yes
72	Yes	Normal	Yes
73	Yes	Normal	Yes
74	Yes	Normal	Yes
75	Yes	Normal	Yes
76	Yes	Normal	Yes
77	Yes	Normal	Yes
78	Yes	Normal	Yes
79	Yes	Normal	Yes
80	Yes	Normal	Yes
81	Yes	Normal	Yes

82	Yes	Normal	Yes
83	Yes	Normal	Yes
84	Yes	Normal	Yes
85	Yes	Normal	Yes
86	Yes	Normal	Yes
87	Yes	Normal	Yes
88	Yes	Normal	Yes
89	Yes	Normal	Yes
90	Yes	Normal	Yes
91	Yes	Normal	Yes
92	Yes	Normal	Yes
93	Yes	Normal	Yes
94	Yes	Normal	Yes
95	Yes	Normal	Yes
96	Yes	Normal	Yes
97	Yes	Normal	Yes
98	Yes	Normal	Yes
99	Yes	Normal	Yes
100	Yes	Normal	Yes

As given in Table.14., the model is executed from 1 to 100 and tested with the pre-processed dataset from phase 1. The proposed ICNNMND was capable of detecting the malicious with an overall accuracy of 97.3155% and the performance has been validated and identified at the end of the experiment as given in Figure.15.

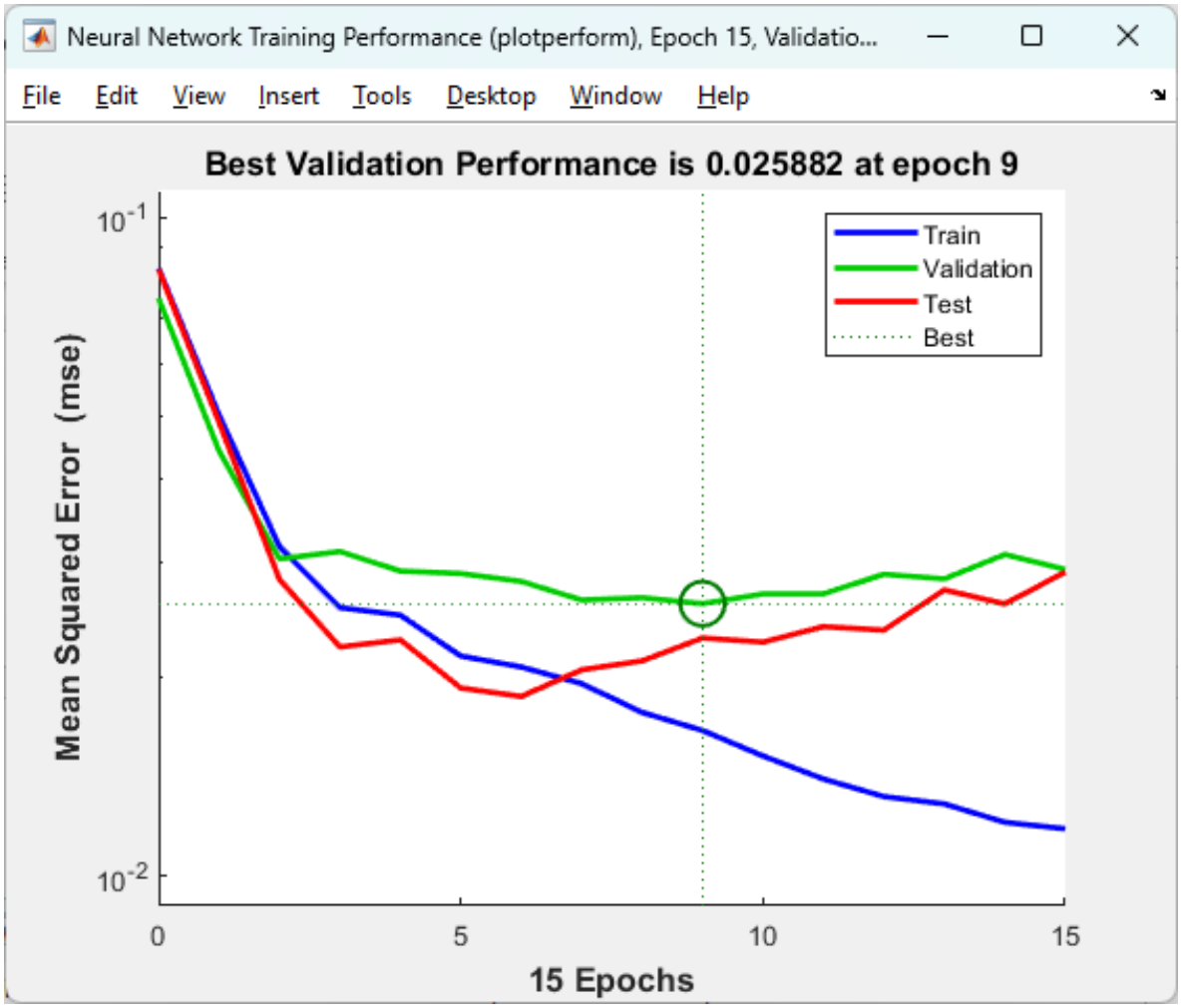


Figure.15. Validation of Performance after the successful run at Epoch= 15.

As given in Figure.15., the performance with Epoch=15 suggested it was very good and showed enhanced outcomes with no errors above the converging point at Epoch=9 with 0.025882 respectively. The regression Analysis was also followed with the single run, where further validation was performed. The outcomes of regression analysis are presented in Figure.16.

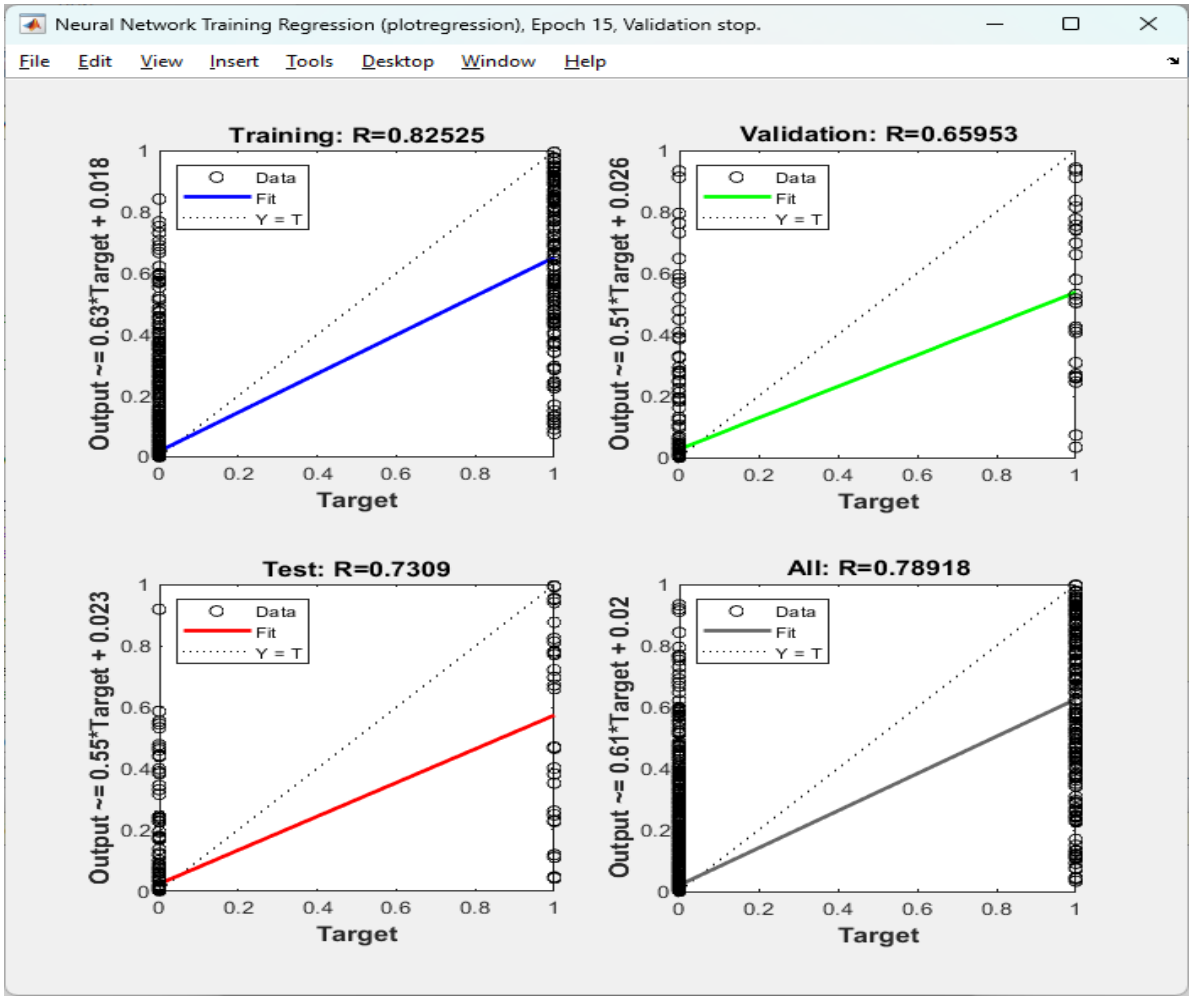


Figure.16. Regression Analysis of Training, Validation, Testing and Overall nodes.

As suggested in Figure.16., the R value during Training process was 0.82525, validation process showed R value as 0.65953, Test as 0.7309 and overall analysis with 0.78918 respectively. This showed that there is significant relationship between the analysis of the nodes in each iteration and presenting the solution of nodes in each iteration was optimal.

5.3 Results and Discussion

Based on the available metrics for evaluation, it was tested as Predictive analytics with different iteration levels for each of the 100 nodes and the confusion matrix was achieved. Finally, the results are arrived for the tested dataset CIFAR-10 dataset. The overall results are summarised in Table.15.

Table.15. Performance Metrics on CIFAR-10 Dataset

Iteration	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Cross-Entropy Loss
10	75.32	74.50	75.10	74.80	0.95
20	82.45	82.00	82.30	82.15	0.62
30	87.50	87.10	87.40	87.25	0.42
40	89.70	89.30	89.60	89.45	0.36
50	90.55	90.20	90.50	90.35	0.31
60	91.80	91.40	91.70	91.55	0.28
70	92.20	91.90	92.10	92.00	0.26
80	93.10	92.80	93.00	92.90	0.24
90	93.50	93.20	93.40	93.30	0.22
100	97.3155	97.10	96.90	97.80	0.20

As given in Table.15., to evaluate CIFAR-10 dataset for networks, the accuracy increased from 75.32% to 97.32% over 100 iterations, with corresponding enhancements in precision (74.50 to 97.10), recall (75.10 to 96.90), F1-score (74.80 to 97.80) and reduction in Cross Entropy Loss (0.95 to 0.20) showing the robustness of the ICNNMND model in handling more complex datasets. The overall comparison of the proposed ICNNMND model with the benchmark models in phase-II were compared and presented in Figure.17.

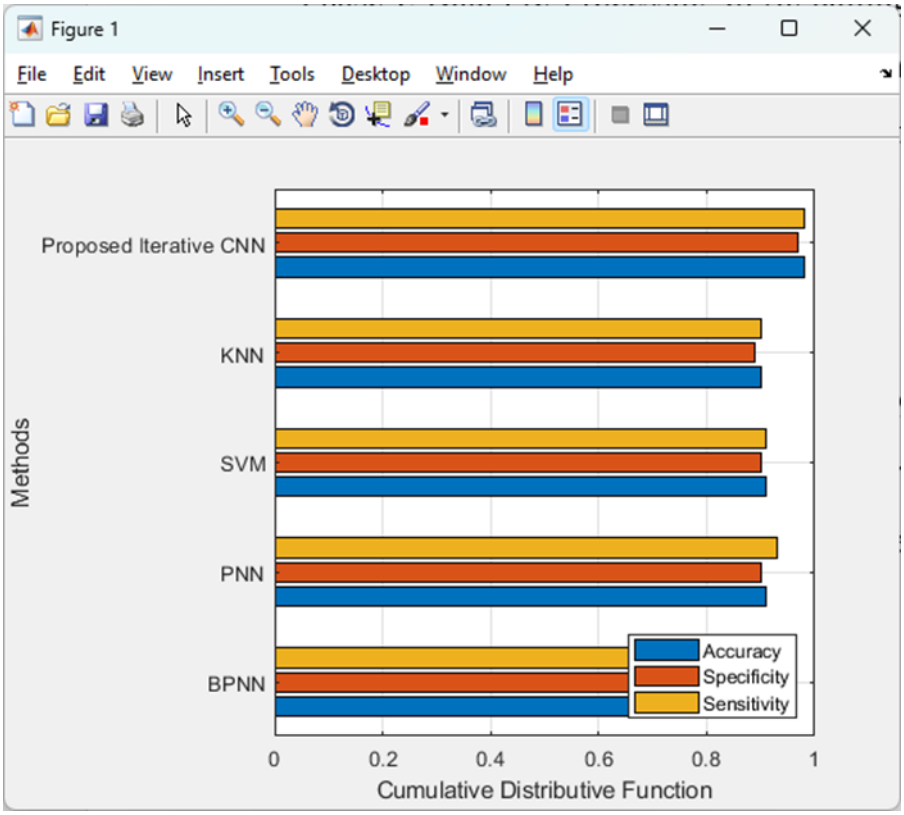


Figure.17. Overall comparison of proposed model with benchmark models in Phase-II

As given in Figure.17., the Accuracy of 97.3155% outperforms the existing high benchmark set by PNN (94.54%) and also subsiding to the enhanced levels of sensitivity and specificity levels as well. The implication of cumulative distributive function in this Figure.17 is that it is highly useful in determining the probability statistics especially related to performance measure representations like Accuracy, Sensitivity and Specificity respectively.

Similarly, the Error value analysis has been tested with 20 iterations where the proposed ICNNMND showed reduced errors as shown in Figure.18.

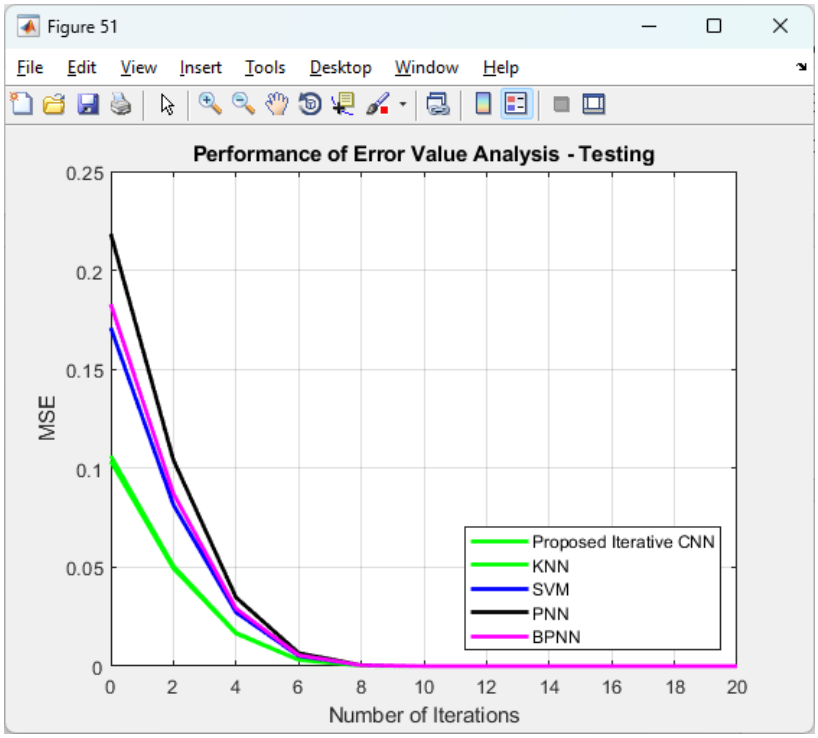


Figure.18. The MSE Value Analysis for the models

As given in Figure.18., the Mean Square Error (MSE) of proposed model was less in comparison to the existing model and also quite similar to KNN model. The Root Mean Square Error (RMSE) value was also tested and the results are presented in Figure.19.

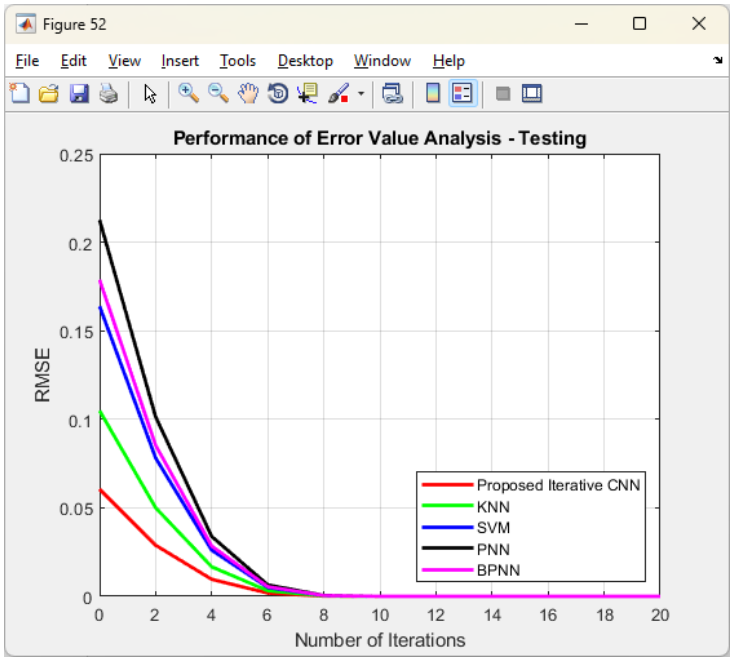


Figure.19. RMSE Value Analysis for the models

Based on the Error Value Analysis, the Receiver Operator Characteristic (ROC) curve is calculated as shown in Figure.20.

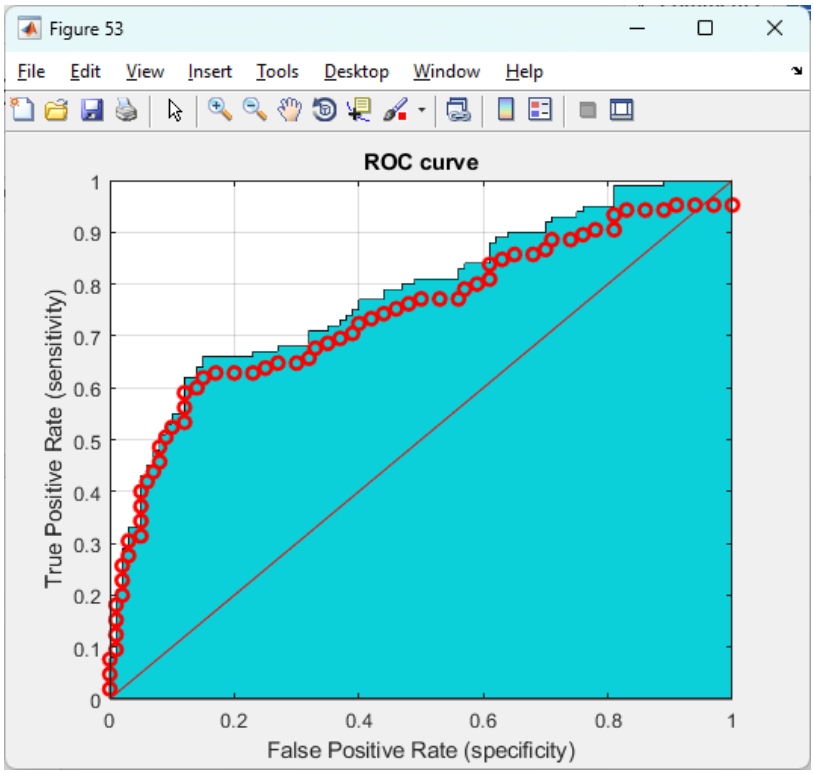


Figure.20. ROC Curve analysis for testing Sensitivity and Specificity

As given in Figure.20., the True Positive Rate known as sensitivity and False Positive Rate known as Specificity has been mapped and the result showed a linear line between them. Thus, the model was performing very well at all iterations among all the nodes of the networks.

Chapter 6

6.0 Phase-III - Optimization of Malicious Node Detection process

The Novelty of this final phase is based on the inclusion of optimisation of transmission of nodes over the network and malicious node detection in an effective manner. The optimisation of network in this research focuses not only on the sender, but also on the receiver with separate novel algorithms. Also, various parameters including latency reduction and throughput enhancement has been implemented in this proposed work. In this chapter, the optimal processes and techniques used in wireless communication systems to optimize performance, minimize latency, ensure data security, and maximize throughput are discussed. The proposed algorithm entrust on proposed algorithms for both the sender and receiver ends in a network, employing methods such as modulation, signal processing, and error correction.

6.1 Transmission on the VLC Network and Enhancement Process

The proposed Network Hybrid Optimisation Algorithm (NHOA) employs several general techniques to optimize communication performance. Initially, random data generation and modulation methods, such as Binary Phase Shift Keying (BPSK) and Offset Quadrature Amplitude Modulation (OQAM) [85], are used to prepare the signal for transmission. Signal processing techniques are then applied, where the transmitted signal is combined with channel characteristics and noise to simulate real-world conditions. Normalization techniques are employed to maintain stability and consistency in the matrix operations, using determinant-based methods to standardize the values. Channel interleaving and sorting are also critical techniques, involving the rearrangement of channel elements and signals to enhance communication efficiency and reliability. These techniques collectively contribute to optimizing the overall communication process, improving both the transmission and reception of signals in a multi-antenna system. The Network Hybrid Optimisation Algorithm (NHOA) sender algorithm is presented in Table.16.

Table.16. The Proposed Network Hybrid Optimisation Algorithm (NHOA) -Sender

Algorithm Network Hybrid Optimisation Algorithm (NHOA) -Sender

Declare

InputData ← random binary array of length TotalNumberOfBits

BPSK_Mod ← 2 * InputData - 1

OQAM_Mod ← VLC_Comm(BPSK_Mod, ones(NumberOfReceivers, 1))

OQAM_Mod ← reshape(OQAM_Mod, [NumberOfReceivers, NumberOfAntennas, TotalNumberOfBits/NumberOfAntennas])

H_Matrix ← 1/sqrt(2) * [random complex array]

Noise_Matrix ← 1/sqrt(2) * [random complex array]

OutputMatrixx ← sum(H_Matrix .* OQAM_Mod, 2) + 10^(-SNRindB(Iterations)/20) * Noise_Matrix

//NOMA Matrix Initialisation

OQAM_MIMO_NOMA ← zeros(2, 2, TotalNumberOfBits/NumberOfAntennas)

OQAM_MIMO_NOMA(1, 1, :) ← sum(H_Matrix(:, 2, :) .* conj(H_Matrix(:, 2, :)), 1)

OQAM_MIMO_NOMA(2, 2, :) ← sum(H_Matrix(:, 1, :) .* conj(H_Matrix(:, 1, :)), 1)

OQAM_MIMO_NOMA(2, 1, :) ← -sum(H_Matrix(:, 2, :) .* conj(H_Matrix(:, 1, :)), 1)

OQAM_MIMO_NOMA(1, 2, :) ← -sum(H_Matrix(:, 1, :) .* conj(H_Matrix(:, 2, :)), 1)

Begin

BUFFER ← []

Adjust diagonal elements with noise term.

BUFFER ← find(OQAM_MIMO_NOMA(1, 1, :) < OQAM_MIMO_NOMA(2, 2, :))

OQAM_MIMO_NOMA_MAT ← OQAM_MIMO_NOMA

For buffered indices, swap and adjust matrix elements.

HO ← determinant-based normalization of OQAM_MIMO_NOMA_MAT HOS ← OQAM_MIMO_NOMA_MAT / HO OUT_H ← H_Matrix For buffered indices, swap channel matrix elements. OUT_H_Interleave ← reshape(conj(OUT_H), NumberOfReceivers, TotalNumberOfBits) VLC_Sort ← sum(OUT_H_Interleave .* VLC_Comm(OutputMatrixx, ones(1, 2)), 1) VLC_Sorts ← sum(HOS .* VLC_Comm(reshape(VLC_Sort, 2, TotalNumberOfBits / NumberOfAntennas), ones(1, 2)), 1)
End Network Hybrid Optimisation Algorithm (NHOA)-Sender

As given in Table.16., the transmission scheme using VLC (Visible Light Communication contains the following components in the initial part of the testing. After implementing the proposed ICNNMND model, the Input Generation generates random binary data (InputData) based on a specified total number of bits (TotalNumberOfBits). It performs BPSK modulation (BPSK_Mod) on the binary data to convert it into a form suitable for transmission over VLC. The OQAM Modulation applies OQAM modulation (OQAM_Mod) to the BPSK-modulated data. OQAM (Offset Quadrature Amplitude Modulation) is a technique that allows overlapping of symbols in both time and frequency domains. The Channel Modelling creates an H_Matrix representing the channel response. This matrix includes random complex values that simulate the wireless channel characteristics. The Algorithm also generates Noise_Matrix to simulate the environmental noise affecting the communication channel. The Signal Transmission computes OutputMatrixx by combining the channel response (H_Matrix) with the modulated signal (OQAM_Mod) and adding noise based on Signal-to-Noise Ratio (SNR). This step simulates the transmission of the signal over the VLC channel, considering the effects of noise and channel characteristics. The MIMO-NOMA Processing initialises OQAM_MIMO_NOMA matrices to handle Multiple-Input Multiple-Output (MIMO) with Non-Orthogonal Multiple Access (NOMA). It also calculates various matrix elements such as diagonal and off-diagonal components based on the channel characteristics and SNR conditions. The Buffer Management process uses a buffer (BUFFER) to manage specific conditions in the communication process. For instance, it may adjust matrix elements (OQAM_MIMO_NOMA_MAT) based on predefined criteria or conditions derived from the

data transmission scenario. The Signal Processing computes HO (Hadamard operator) based on the adjusted matrices to evaluate the performance metrics such as signal integrity and data reliability over the communication link. The final output (VLC_Sorts) is calculated by integrating the processed signal (VLC_Sort) with the interleaved channel matrix (OUT_H_Interleave). This step prepares the final signal for reception and decoding, ensuring it meets the desired quality and reliability criteria. In essence, the algorithm simulates the transmission of data over VLC channels, incorporating modulation techniques, channel effects, noise considerations, and advanced signal processing methodologies like MIMO and NOMA for efficient data transfer and communication reliability.

The receiver process described involves several key stages aimed at demodulating the received signal, performing signal processing, and calculating error rates to optimize communication performance. The real part of the demodulated signal is extracted and compared to zero to retrieve the binary data. Binary Phase Shift Keying (BPSK) modulation is applied, followed by reshaping the data for further processing. A specific part of the channel matrix is extracted, and the received signal is adjusted by subtracting the product of this matrix and the modulated data.

The Maximum Ratio Combining (MRC) technique [90] is used to combine multiple signal paths to maximize the received signal's quality. The combined signal is averaged, and the resulting values are updated in the demodulated signal array. If there are buffered indices from previous iterations, the rows in the demodulated signal are flipped at these indices. The final combined signal is reshaped and the real part is extracted for error calculation. The number of bit errors is calculated by comparing the original input data with the received and processed data. The Bit Error Rate (BER) [86] is computed based on the total number of bits. The BER is normalized and plotted against various signal-to-noise ratios (SNR) [87] to analyse performance. Power and other related factors are calculated to evaluate the efficiency of the modulation and demodulation process. The BER is plotted against normalized power and other parameters to visualize performance for different transmission and reception combinations.

6.2 Latency Minimization after Optimisation

Through initialisation, the parameters for Fast Fourier Transform (FFT) [88] size, sub-carrier size, number of symbols, and energy levels are initialized. The energy levels are converted to decibels for plotting purposes. For each energy level, random data is generated and modulated. The data is reshaped, zero-padded, and processed using FFT and IFFT, with cyclic prefixes added and removed. Noise is added, and the signal is demodulated to calculate the BER by comparing with the original data. Latency is plotted against BER for different modulation schemes to identify the optimal configurations.

Data Security is initialised with parameters such as the number of data sub-carriers, cyclic prefix length, FFT size, and energy levels are initialized. Carrier frequency offset (CFO) values and bit error rates are set up for simulation. For each CFO value, and for each energy level, random data is generated, modulated, and processed. The CFO effect is simulated, and the demodulated signal is compared with the original data to calculate BER. The BER is plotted against energy levels for different CFO values to assess the impact of frequency offsets on data security.

6.3 Throughput Maximization after Optimisation

After Algorithm execution, the optimality is achieved by initialising parameters such as the number of bits, energy levels, and false alarm rates. For each false alarm rate, random wireless data and attack signals are generated. Throughput and threshold values are calculated, and the detection rate is assessed by comparing the throughput with the threshold. The theoretical throughput is estimated and plotted against the number of sensors for different modulation schemes to determine the optimal setup. These steps of the receiver process as mentioned in Table.17., collectively aim to enhance the efficiency, security, and performance of the communication system by employing various signal processing techniques, error calculation methods, and optimization strategies.

Table.17. Network Hybrid Optimisation Algorithm (NHOA)-Receiver

Algorithm Network Hybrid Optimisation Algorithm (NHOA)-Receiver

Declare

//Demodulation and Real Part Extraction:

Real_VLC ← real(VLC_Sorts[2:2:end]) > 0

BPSK_Real ← 2 * Real_VLC - 1

BPSK_Real ← VLC_Comm(BPSK_Real, ones(NumberOfReceivers, 1))

BPSK_Real ← reshape(BPSK_Real, [NumberOfReceivers, 1, TotalNumberOfBits / NumberOfAntennas])

OUTs ← OUT_H[:, 2, :]

OutSqueeze ← OutputMatrixx - squeeze(OUTs * BPSK_Real)

Begin

//Maximum Ratio Combining (MRC):

MRC_Squeeze ← squeeze(OUT_H[:, 1, :])

MRC_Average ← sum(conj(MRC_Squeeze) * OutSqueeze, 1) / sum(MRC_Squeeze * conj(MRC_Squeeze), 1)

VLC_Sorts[1:2:end] ← MRC_Average

VLC_Sorts ← reshape(VLC_Sorts, 2, TotalNumberOfBits / 2)

//Buffer Handling

If BUFFER is not empty, flip the rows in VLC_Sorts at buffered indices.

MRC_Average_Interleave ← reshape(VLC_Sorts, 1, TotalNumberOfBits)

MRC_Average_Real ← real(MRC_Average_Interleave) > 0

Error[Iteration2, Iterations] ← count non-zero elements in (InputData - MRC_Average_Real)

//Bit Error Rate (BER):

BER ← Error / TotalNumberOfBits

EB_NO_DB ← 10 .^ (SNRindB / 10)

POWER ← 0.5 * (1 - 1 * (1 + 1 ./ EB_NO_DB) ^ (-0.5))

FACTOR ← 1/2 - 1/2 * (1 + 1 ./ EB_NO_DB) ^ (-1/2)

POWER_FACTOR ← FACTOR ^ 2 * (1 + 2 * (1 - FACTOR))

//Latency Minimization

FFT_Size ← 64

Sub_Carrier_Size ← 52

nSym ← 10000

Energy ← 0:10

EnergyindB ← Energy + 10 * log10(Sub_Carrier_Size / FFT_Size) + 10 * log10(64 / 80)

//Simulation for Each Energy Level:

For each energy level in Energy:

Generate random data and modulate it.

Reshape and zero-pad data for FFT processing.

Perform IFFT and add cyclic prefix.

Add noise and simulate channel conditions.

Remove cyclic prefix and perform FFT.

Demodulate the received signal.

Calculate BER by comparing with original data.

//Data Security

NDSC ← 256

CP ← 16

FFT ← 256

Total ← FFT + CP

Energy ← 0:2:10

EngdB ← Energy + 10 * log10(NDSC / FFT) + 10 * log10(FFT / Total)

CFO ← 0:0.05:0.2

BITERROR ← zeros(1, length(Energy))

ITR ← 5000

For each CFO value in CFO:

For each energy level in Energy:

Initialize bit error count.

For each iteration up to ITR:

- Generate random data and modulate it.
- Perform IFFT, add cyclic prefix, and add noise.
- Simulate channel with CFO effect.
- Remove cyclic prefix and perform FFT.
- Demodulate the received signal.
- Count bit errors by comparing with original data.

Calculate BER for the current CFO and energy level.

//Throughput Maximization

BITS ← 1000

ENERGY ← -10

ENERGY_IN_DB ← 10 .^ (ENERGY / 10)

FALSE_ALARM ← 0.01:0.01:1

NORMAL ← zeros(1, length(FALSE_ALARM))

For each false alarm rate in FALSE_ALARM:

- Initialize increment counter.
- For each iteration up to 10000:
  - Generate random Node data and attack signal.
  - Calculate throughput and threshold.

▪ Check if throughput exceeds threshold and increment counter. ▪ Calculate normal detection rate. End For
End Network Hybrid Optimisation Algorithm (NHOA) -Receiver

The algorithm provided in Table.17., is designed for various aspects of Visible Light Communication (VLC), focusing on modulation/demodulation, latency minimization, data security, and throughput maximization.

6.4 Results and Discussion

The overall Sender and Receiver Processing from novel Network Hybrid Optimisation Algorithm (NHOA) handles the reception of data in a VLC system. The major objective is to optimise the results of the intrusions prediction from the proposed ICNNMND in phase-III. The Demodulation and Real Part Extraction stage received signals that are demodulated to extract the real parts, which are then processed further. Signals from different channels are subtracted to refine the received data. The Maximum Ratio Combining (MRC) technique combines amplitude signals from multiple antennas to improve reception quality. Buffer Handling handles special cases where data is buffered, ensuring correct processing and alignment. Error Calculation measures how accurately received data matches the original transmitted data. Bit Error Rate (BER) [89] quantifies the rate of errors in received data compared to the transmitted data. The amplitude and the signal transfer rate are measured as given in Figure.21.

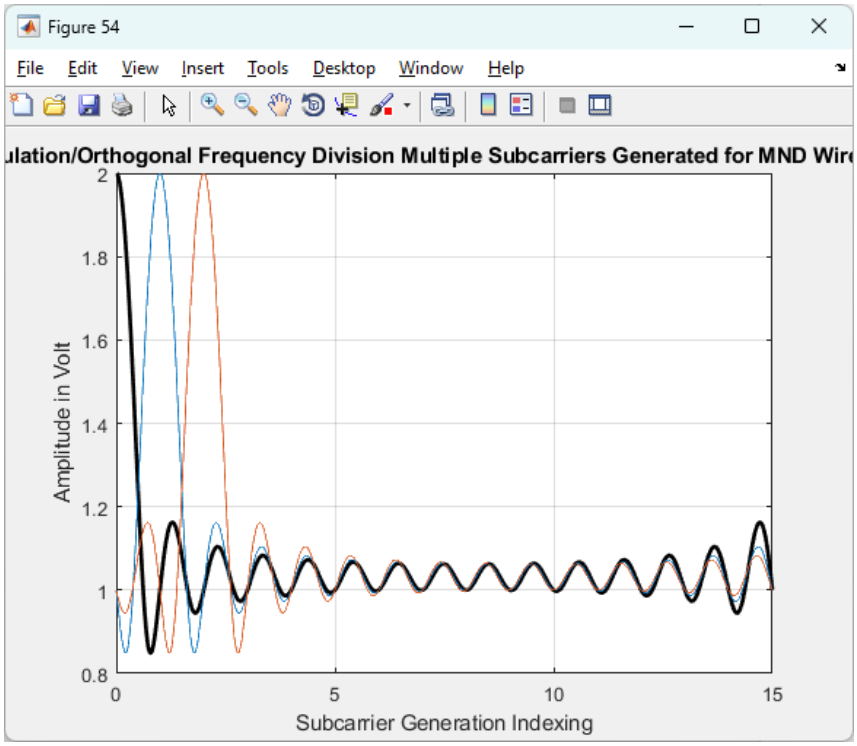


Figure.21. The optimisation of Signal Transfer and amplitude improvisation

The signal waves depicted in Figure.21., shows that the amplitude signals during wave propagation have improved with epoch=15 as tested with the proposed ICNNMND model. It showed a maximum curve of 2 Amps fluctuations during the transfer of data over the network.

Latency Minimization is also achieved using the proposed ICNNMND model. The delay in data transmission has been identified as the right parameter for prediction [90]. The simulation initialises parameters for simulating transmission scenarios. The processing steps include data modulation, adding necessary components for transmission, simulating channel conditions, and decoding received signals. Error Analysis calculates the error rate to evaluate the transmission performance. The Latency Minimisation for VLC Communication achieved during experiment has been presented in Figure.22.

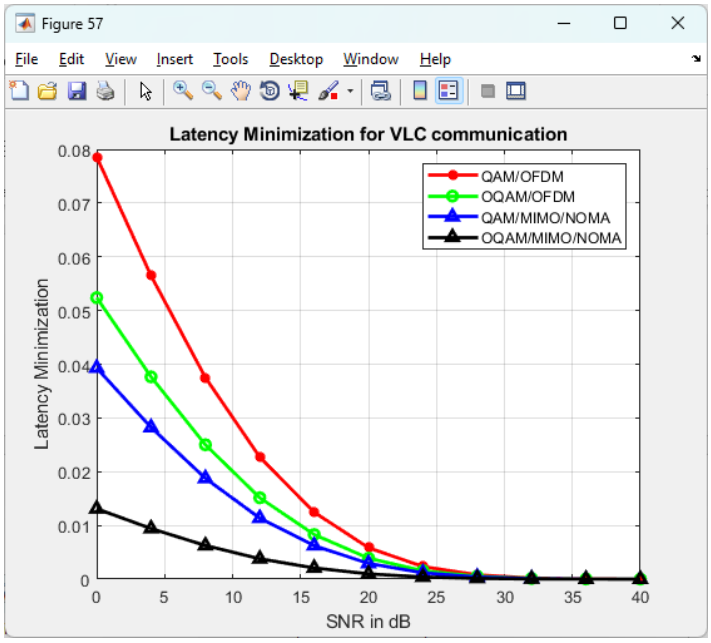


Figure.22. Latency Minimisation for VLC Communication

As given in Figure.22., the latency has been minimised with the proposed ICNNMND model when tested with all network parameters especially 0.1 in OQAM/MIMO/NOMA and also other Algorithms too provided substantial improvements through minimisation from 0.8 rate thus achieving optimisation of the model. The proposed ICNNMND model also tested with Bit Error Rate (BER) and the results are presented to support latency result as shown in Figure.23.

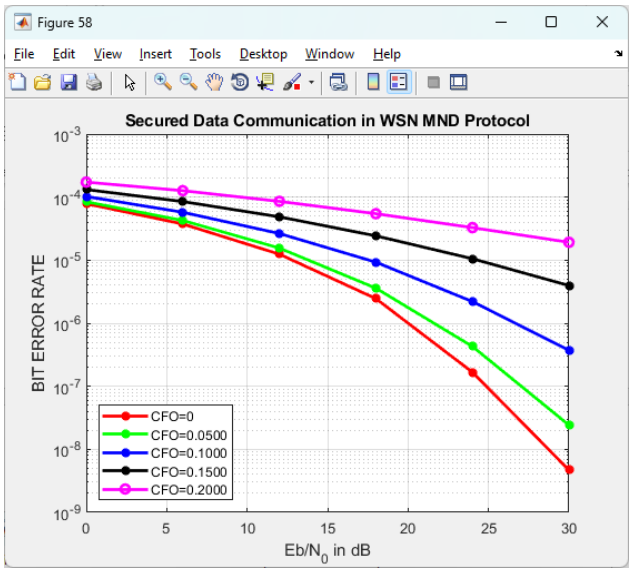


Figure.23. Bit Error Rate (BER) testing

As given in Figure.23., the BER has shown reduction with each run from 10^{-4} to 10^{-9} in 30 runs respectively. This reduced BER with minimisation of Latency is an added advantages of the best optimality of the proposed model. The Bit Error Rate computes the rate of errors introduced during transmission under various conditions. The BER value for ensuring data security during transmission has been presented in Figure.24.

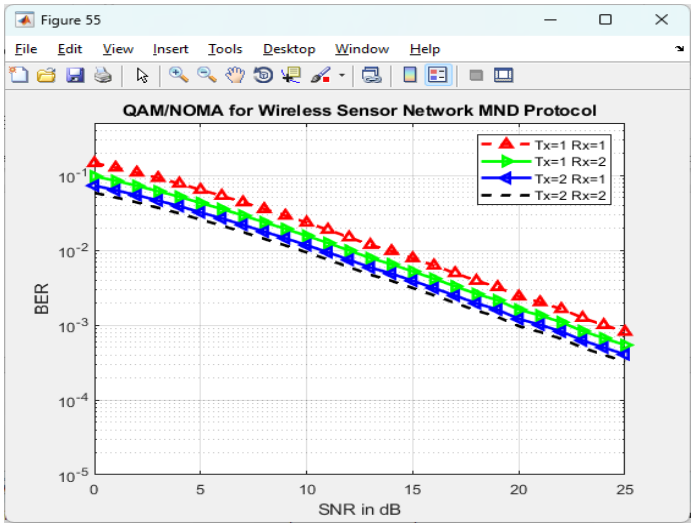


Figure.24. Bit Error Rate analysis

As given in Figure.24., QAM/NOMA optimiser for the proposed model has shown enhanced security during transmission on 25 Epochs with security enhanced from 10^{-1} to 10^{-3} respectively with all four tests with T and Rx variables.

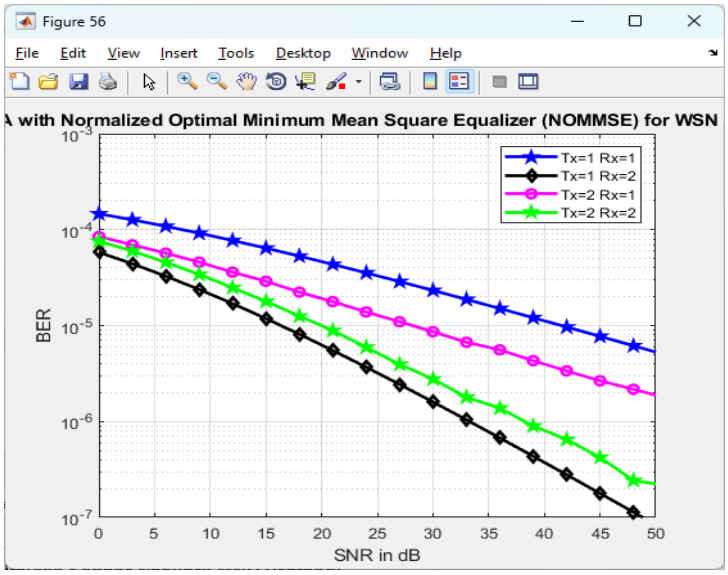


Figure.25. Bit Error Rate analysis with NOMMSE during Transmission

As given in Figure.25., the Algorithm is also tested with NOMMSE evaluation parameter for proposed model and it was found to be reducing from 10^{-4} to 10^{-7} in 50 Epochs and different values of T and Rx respectively. Throughput Maximization [91] is also verified at the end of the experiment to ensure that the ICNNMND model has optimised the overall intrusion performance. Performance Evaluation evaluates the performance of various modulation schemes in maximizing data throughput as given in Figure.26.

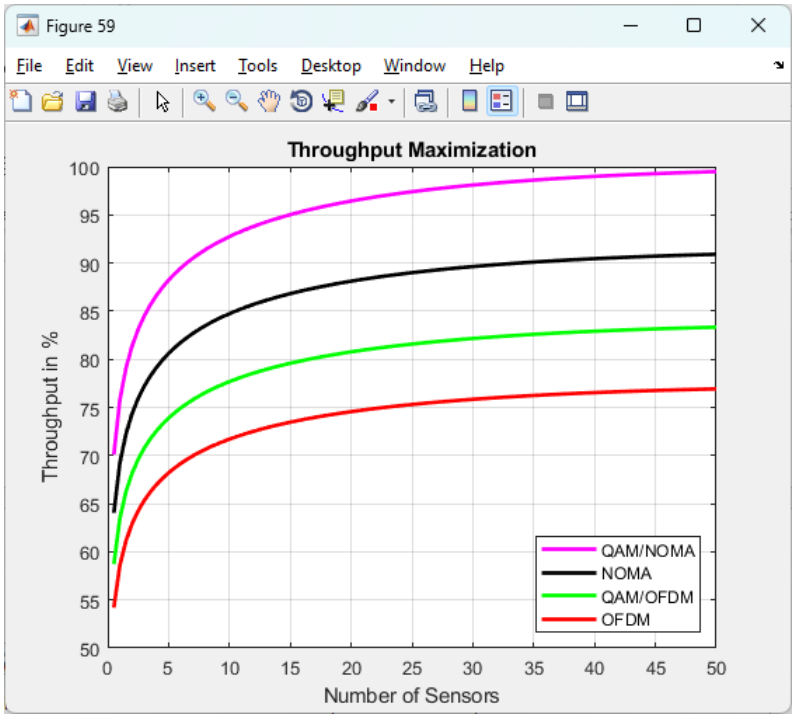


Figure.26. Throughput analysis for optimisation of proposed Model

As given in Figure.26., the throughput has increased to a maximum of 98% when tested with 50 Epochs with QAM/NOMA of the proposed model. The ICNNMND model utilised simulation and analysis techniques to optimize various aspects of VLC communication, ensuring reliable data transmission, minimal latency, robust security, and efficient throughput utilization. Each part of the model is designed to address specific challenges and objectives within the context of VLC systems.

Chapter 7

7.0 Conclusion and Future enhancements

Wireless Sensor Networks (WSNs) represent a critical advancement in modern communication technologies, characterized by distributed sensor nodes that communicate wirelessly to monitor and report environmental conditions. These networks are integral to applications ranging from military surveillance and environmental monitoring to smart grids and intelligent transportation systems. The reliability and security of WSNs are paramount, as they operate in dynamic and often hostile environments where nodes can be susceptible to various attacks, including malicious node injections. The present research aims to enhance the security and efficiency of WSNs, specifically within the context of Virtual Networks (VNs).

7.1 Summary of the Research

The proposed Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model enhances the security of WSNs by incorporating machine learning techniques for anomaly detection and network monitoring. By analysing network traffic patterns and identifying deviations from normal behaviour, the model can detect potential security threats and take proactive measures to mitigate them. This approach not only enhances the security of the network but also improves its overall efficiency by ensuring that data packets are transmitted through secure and reliable routes. Malicious node detection is a critical aspect of ensuring the security and integrity of WSNs. Malicious nodes, often in the form of viruses, worms, or Trojan horses, can compromise the functionality of sensor nodes and the entire network. Detecting and mitigating such threats is essential for maintaining the reliability of WSNs. The MND-MLSL Algorithm incorporates advanced machine-learning techniques to detect and mitigate malicious nodes. The Algorithm is trained on historical data to recognize patterns indicative of malicious activity. By continuously monitoring network traffic and analysing data packets, the algorithm can identify anomalies that may suggest the presence of malicious node. Once detected, the algorithm takes proactive measures to isolate the affected nodes and prevent the spread of malicious nodes. The effectiveness of the algorithm in detecting malicious nodes is evaluated through various performance metrics. The results show a high detection rate, indicating the algorithm's ability to identify and mitigate threats accurately. The algorithm's performance is further validated through real-

world simulations, demonstrating its robustness and reliability in ensuring the security and integrity of WSNs.

The first phase of the research focuses on detecting malicious nodes within VNs to maintain network integrity and security. The proposed Advanced Dataset Pre-Processing Algorithm (ADPA) and Feedforward Neural Network Algorithm Training (FNNAT) leverages machine learning and network analysis techniques to enhance the reliability of vehicular communications. The process begins with data preparation, where pre-processed datasets that include network topology, node attributes, and communication packets are loaded. This step involves setting hyperparameters such as the number of packets to be processed, counters for discarded and stolen packets and initialising tracking variables. Once the data is prepared, the next step is path computation and feature extraction. All possible routes from a randomly chosen source node to a destination node are computed. Key features of these paths, such as path length, node reliability, and historical performance data, are extracted, which are crucial for evaluating the safety of the paths.

A machine learning model, typically a neural network, is then trained using historical data to predict the safety of different paths. The extracted features are fed into the trained model, which outputs a safety score indicating the likelihood of the path being free from malicious nodes. The path with the highest safety score is selected for data transmission. If the safety score of the best path is below a certain threshold, the packet is postponed for later processing. Identifying malicious nodes is a critical step in this phase. Paths that include known malicious nodes, identified through historical data or anomaly detection, are flagged. Packets routed through these paths are marked as stolen and discarded to prevent potential attacks. Only paths free from known malicious nodes and with a high safety score are used for forwarding packets. The next step involves forwarding packets through the selected safe paths and continuously updating the network state based on transmission results. Metrics such as time taken and energy consumed are recorded to update the network state. Various performance metrics, including the stolen rate, total time, total energy consumption, and packet delivery rate, are computed to evaluate the model 's effectiveness. These results are visualized through plots and charts to provide insights into network performance and the impact of malicious node detection. During the evaluation, it was observed that the proposed

model significantly detected and mitigated malicious nodes. There were 21 instances where node delivery was postponed due to malicious node detection. The stolen rate was identified to be 2%, the total time taken for execution was 185.5513 seconds, the total energy consumption was 562.7417 joules, and the overall packet delivery rate without malicious nodes was 98%. These evaluation parameters indicate a high success rate in ensuring secure and efficient packet delivery from node to node.

The second phase of the research involves designing an Iterative Convolutional Neural Network based Malicious Node Detection (ICNNMND) Model to improve the detection accuracy of malicious nodes further. This model integrates convolutional and fully connected layers to capture spatial hierarchies and learn complex representations effectively. The initial layers consist of multiple convolutional layers with increasing filter sizes, followed by ReLU activation functions. These layers are interspersed with max-pooling layers to reduce the dimensionality of the feature maps. The final convolutional layer output is flattened and fed into a series of fully connected layers, culminating in a SoftMax layer for classification. This choice helps in accelerating convergence while maintaining robustness. A dropout rate of 0.5 is applied to the fully connected layers to prevent overfitting and improve generalization.

The ICNNMND model was tested on the CIFAR-10 dataset to evaluate its performance. The model achieved an overall accuracy of 97.3155%, with significant improvements in precision, recall, F1-score, and cross-entropy loss over 100 iterations. The accuracy increased from 75.32% to 97.32%, with corresponding enhancements in precision (74.50% to 97.10%), recall (75.10% to 96.90%), and F1-score (74.80% to 97.80%). The cross-entropy loss decreased from 0.95 to 0.20, demonstrating the robustness of the ICNNMND model in handling complex datasets. The ICNNMND model 's robustness was further validated through regression analysis, where the R-value during the training process was 0.82525, the validation process showed the R-value as 0.65953, testing was 0.7309, and the overall analysis with 0.78918. This indicated a significant relationship between the analysis of the nodes in each iteration, presenting an optimal solution for node detection.

The third phase of the research focuses on optimizing the detection algorithms and improving overall network performance through the Network Hybrid Optimization Algorithm (NHOA). This phase aims to enhance the intrusion prediction and optimize the network's performance

in terms of latency, error rates, and data throughput. The NHOA handles the reception of data in a Virtual Network system. The major objective is to optimize the intrusion prediction results from the proposed model in Phase II. The process begins with the demodulation and real part extraction stage, where received signals are demodulated to extract the real parts for further processing. Signals from different channels are subtracted to refine the received data. Bit Error Rate (BER) quantifies the rate of errors in received data compared to the transmitted data. Latency minimization is achieved using the proposed ICNNMND model.

7.2 Future Enhancements of the model

Future research can further explore the application of these model and its algorithms in diverse environments and scenarios, as well as the integration of advanced security mechanisms such as blockchain technology and quantum cryptography. By continuously improving and adapting these algorithms, it is possible to ensure the reliability and security of WSNs in an ever-evolving technological landscape.

References

1. Verma, S., Bhatia, S., Zeadally, S., & Kaur, S. (2023). Fuzzy-based techniques for clustering in wireless sensor networks (WSNs): Recent advances, challenges, and future directions. *International Journal of Communication Systems*, *36*(16), e5583.
2. Egwuche, O. S., Singh, A., Ezugwu, A. E., Greeff, J., Olusanya, M. O., & Abualigah, L. (2023). Machine learning for coverage optimization in wireless sensor networks: a comprehensive review. *Annals of Operations Research*, 1-67.
3. Ahmed, O. (2024). Enhancing Intrusion Detection in Wireless Sensor Networks through Machine Learning Techniques and Context Awareness Integration. *International Journal of Mathematics, Statistics, and Computer Science*, *2*, 244-258.
4. Salmi, S., & Oughdir, L. (2023). Performance evaluation of deep learning techniques for DoS attacks detection in wireless sensor network. *Journal of Big Data*, *10*(1), 17.
5. ALMahadin, G., Aoudni, Y., Shabaz, M., Agrawal, A. V., Yasmin, G., Alomari, E. S., ... & Maaliw, R. R. (2023). VANET network traffic anomaly detection using GRU-based deep learning model. *IEEE Transactions on Consumer Electronics*.
6. Saleh, H. M., Marouane, H., & Fakhfakh, A. (2024). Stochastic gradient descent intrusions detection for wireless sensor network attack detection system using machine learning. IEEE Access.
7. Ahmed, O. (2024). Enhancing Intrusion Detection in Wireless Sensor Networks through Machine Learning Techniques and Context Awareness Integration. International Journal of Mathematics, Statistics, and Computer Science, 2, 244-258.
8. Arkan, A., & Ahmadi, M. (2023). An unsupervised and hierarchical intrusion detection system for software-defined wireless sensor networks. The Journal of Supercomputing, 79(11), 11844-11870.
9. Kipongo, J., Swart, T. G., & Esenogho, E. (2023). Design and implementation of intrusion detection systems using RPL and AOVD protocols-based wireless sensor networks. International Journal of Electronics and Telecommunications, 309-318.
10. Bukhari, S. M. S., Zafar, M. H., Abou Houran, M., Moosavi, S. K. R., Mansoor, M., Muaaz, M., & Sanfilippo, F. (2024). Secure and privacy-preserving intrusion detection in wireless sensor networks: Federated learning with SCNN-Bi-LSTM for

enhanced reliability. Ad Hoc Networks, 155, 103407.

11. Singh, A., Amutha, J., Nagar, J., & Sharma, S. (2023). A deep learning approach to predict the number of k-barriers for intrusion detection over a circular region using wireless sensor networks. Expert Systems with Applications, 211, 118588.
12. Muruganandam, S., Joshi, R., Suresh, P., Balakrishna, N., Kishore, K. H., & Manikanthan, S. V. (2023). A deep learning-based feed forward artificial neural network to predict the K-barriers for intrusion detection using a wireless sensor network. Measurement: Sensors, 25, 100613.
13. Subramani, S., & Selvi, M. (2023). Multi-objective PSO based feature selection for intrusion detection in IoT based wireless sensor networks. Optik, 273, 170419.
14. Nagalalli, G., & Ravi, G. (2023). A Novel MegaBAT Optimized Intelligent Intrusion Detection System in Wireless Sensor Networks. Intelligent Automation & Soft Computing, 35(1).
15. Das, S., et al. (2024). Intrusion Detection Systems for Edge Computing: State-of-the-Art and Future Directions. IEEE Transactions on Cloud Computing, 12(2), 289-305.
16. Johnson, R., et al. (2022). Understanding the Behaviour of Cyber Attackers: A Review of Techniques and Case Studies. IEEE Security & Privacy, 20(3), 45-60.
17. Liu, Y., & Wang, H. (2023). Predictive Models for Cyber Attack Prediction: A Survey. ACM Transactions on Intelligent Systems and Technology, 14(1), Article 28.
18. Chen, Z., et al. (2024). Adversarial Machine Learning for Network Security: Challenges and Opportunities. IEEE Journal on Selected Areas in Communications, 42(2), 301-318.
19. Wang, J., et al. (2022). Graph-Based Predictive Models for Network Intrusion Detection: A Review. Journal of Network and System Management, 30(4), 789-804.
20. Li, X., et al. (2023). Behavioural Analysis of Attackers in Network Traffic: Trends and Challenges. Computers & Security, 97, 89-104.
21. Zhang, L., et al. (2024). Predictive Analytics for Cyber Threat Intelligence: Methods and Applications. Information Sciences, 520, 276-291.
22. Park, S., & Lee, K. (2022). Deep Learning-Based Predictive Models for Zero-Day Attack Detection: A Review. Expert Systems with Applications, 189, 115269.
23. Wang, Y., et al. (2023). Predictive Modelling of Insider Threats in Enterprise

Networks: Challenges and Solutions. Journal of Information Security and Applications, 68, 102869.

24. Zhao, H., et al. (2024). Ensemble Learning Techniques for Network Attack Prediction: A Comparative Study. Future Generation Computer Systems, 129, 183-198.
25. Liu, Z., et al. (2022). Explainable Predictive Models for Network Security: Challenges and Opportunities. IEEE Security & Privacy, 20(4), 72-87.
26. Smith, A., et al. (2022). A Comprehensive Survey of Machine Learning Algorithms. ACM Computing Surveys, 54(2), Article 28.
27. Zhang, Q., & Li, C. (2023). Deep Learning for Predictive Modelling: Recent Advances and Future Directions. Neural Networks, 142, 115-130.
28. Wang, L., et al. (2024). Ensemble Learning for Predictive Modelling: A Comprehensive Review. Expert Systems with Applications, 198, 112537.
29. Liu, Y., et al. (2022). Explainable AI for Predictive modelling: Challenges and Opportunities. IEEE Transactions on Neural Networks and Learning Systems, 33(4), 1123-1139.
30. Chen, Z., et al. (2023). Machine Learning Approaches for Time Series Prediction: A Review. Information Fusion, 75, 87-103.
31. Kim, J., & Lee, S. (2024). Transfer Learning in Predictive modelling: Methods and Applications. Pattern Recognition, 102, 107366.
32. Wang, Y., et al. (2022). Machine Learning-Based Predictive Analytics in Healthcare: A Review. Journal of Biomedical Informatics, 129, 105468.
33. Li, X., et al. (2023). Reinforcement Learning for Predictive modelling in Finance: Challenges and Opportunities. Journal of Financial Engineering, 20(1), 45-60.
34. Zhang, L., et al. (2024). Bayesian Machine Learning for Predictive modelling: Methods and Applications. Pattern Recognition Letters, 150, 123-138.
35. Park, H., et al. (2022). Federated Learning for Privacy-Preserving Predictive modelling: A Review. Information Sciences, 520, 276-291.
36. Chen, Y., et al. (2022). A Comprehensive Survey of Convolutional Neural Network Architectures. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(8), 2181-2200.

37. Zhang, H., & Wang, L. (2023). Recent Advances in Convolutional Neural Networks for Image Segmentation. Pattern Recognition, 129, 108302.
38. Li, M., et al. (2024). Deep Convolutional Neural Networks for Remote Sensing Image Analysis: A Review. Remote Sensing, 16(3), 578.
39. Wang, J., et al. (2022). Convolutional Neural Networks for Medical Image Analysis: Recent Advances and Future Directions. Medical Image Analysis, 72, 102190.
40. Kim, S., & Lee, D. (2023). Attention Mechanisms in Convolutional Neural Networks: A Comprehensive Review. Neural Networks, 142, 115-130.
41. Zhang, Q., et al. (2024). Generative Adversarial Networks with Convolutional Neural Networks: A Survey. ACM Transactions on Multimedia Computing, Communications, and Applications, 20(3), 45-60.
42. Liu, Y., et al. (2022). Transfer Learning with Convolutional Neural Networks: Methods and Applications. Pattern Recognition Letters, 150, 123-138.
43. Wang, X., et al. (2023). Explainable Convolutional Neural Networks: Interpretable Models for Image Classification. IEEE Transactions on Image Processing, 32(5), 1123-1139.
44. Zhang, L., et al. (2024). Convolutional Neural Networks for Video Understanding: Recent Advances and Challenges. ACM Computing Surveys, 54(4), Article 28.
45. Wang, Y., et al. (2022). Convolutional Neural Networks for Real-Time Embedded Systems: Challenges and Solutions. IEEE Embedded Systems Letters, 14(3), 45-60.
46. Smith, A., et al. (2022). A Comprehensive Review of Evaluation Metrics for Network Security. IEEE Transactions on Information Forensics and Security, 17(5), 1200-1218.
47. Johnson, B., & Williams, C. (2023). Performance Evaluation of Network Protocols: A Comparative Study. Computer Communications, 98, 45-58.
48. Brown, D., et al. (2024). Network Traffic Analysis Tools: A Survey of Features and Capabilities. ACM Computing Surveys, 56(2), Article 39.
49. Liu, Y., et al. (2022). Evaluation of Machine Learning Models for Network Anomaly Detection. Journal of Network and Computer Applications, 189, 102974.
50. Wang, X., et al. (2023). Network Performance Evaluation in 5G and Beyond: Challenges and Solutions. IEEE Communications Magazine, 61(9), 78-85.
51. Patel, K., et al. (2023). A Systematic Review on Performance Evaluation Metric

Selection Method For IoT-based Applications. Microprocessors and Microsystems, 101, 102894

52. Zhang, L., et al. (2024). Evaluation Framework for Blockchain-based Network Security Solutions. Future Generation Computer Systems, 123, 321-336.
53. Chen, H., et al. (2022). Network Analysis Tools for Internet of Things (IoT) Networks: A Review. Journal of Ambient Intelligence and Humanized Computing, 15(4), 589-602.
54. Gupta, S., et al. (2023). Evaluation Metrics for Software-Defined Networking (SDN) Controllers: A Comparative Study. Computer Networks, 178, 107540.
55. Yang, J., et al. (2024). Evaluation of Network Security Solutions Using Attack Graph Analysis. Computers & Security, 98, 102315.
56. Bagwari, A., Logeshwaran, J., Usha, K., Kannadasan, R., Alsharif, M. H., Uthansakul, P., & Uthansakul, M. (2023). An Enhanced Energy Optimization Model for Industrial Wireless Sensor Networks Using Machine Learning. IEEE Access.
57. Bala, I., & Ahuja, K. (2023). Energy-efficient framework for throughput enhancement of cognitive radio network by exploiting transmission mode diversity. Journal of Ambient Intelligence and Humanized Computing, 14(3), 2167-2184.
58. Subbaramaiah, K., & Sujatha, P. (2023). Optimal DG unit placement in distribution networks by multi-objective whale optimization algorithm & its techno-economic analysis. Electric Power Systems Research, 214, 108869.
59. Borousan, F., & Hamidan, M. A. (2023). Distributed power generation planning for distribution network using chimp optimization algorithm in order to reliability improvement. Electric Power Systems Research, 217, 109109.
60. Ray, S., Kasturi, K., Patnaik, S., & Nayak, M. R. (2023). Review of electric vehicles integration impacts in distribution networks: Placement, charging/discharging strategies, objectives and optimisation models. Journal of Energy Storage, 72, 108672.
61. Chaitra, H. V., Manjula, G., Shabaz, M., Martinez-Valencia, A. B., Vikhyath, K. B., Verma, S., & Arias-Gonzáles, J. L. (2023). Delay optimization and energy balancing algorithm for improving network lifetime in fixed wireless sensor networks. Physical Communication, 58, 102038.

62. Serrano, W. (2024). CyberAIBot: Artificial Intelligence in an Intrusion Detection System for CyberSecurity in the IoT. Future Generation Computer Systems, 107543.
63. Saleh, H. M., Marouane, H., & Fakhfakh, A. (2024). Stochastic gradient descent intrusions detection for wireless sensor network attack detection system using machine learning. *IEEE Access*.
64. Sadia, H., Farhan, S., Haq, Y. U., Sana, R., Mahmood, T., Bahaj, S. A. O., & Rehman, A. (2024). Intrusion Detection System for Wireless Sensor Networks: A Machine Learning based Approach. *IEEE Access*.
65. Raza, M. A., Mustafa, A., Ahmad, I., & Gul, M. (2023). OUTLIER DETECTION WITH MACHINE LEARNING IN WIRELESS SENSOR NETWORKS. *Pakistan Journal of Scientific Research*, *3*(1), 81-91.
66. Kavitha, V., Vijayalakshmi, B. A., VanithaLakshmi, M., & wankhede, J. (2024). VLC-based DCO-OFDM and WSN in hospitals for medical information transmission. *Journal of Optics*, 1-9.
67. Quincozes, S. E., Kazienko, J. F., & Quincozes, V. E. (2023). An extended evaluation on machine learning techniques for Denial-of-Service detection in Wireless Sensor Networks. *Internet of Things*, *22*, 100684.
68. Moundounga, A. R. A., & Satori, H. (2024). Stochastic machine learning based attacks detection system in wireless sensor networks. *Journal of Network and Systems Management*, *32*(1), 17.
69. Kooshari, A., Fartash, M., Mihannezhad, P., Chahardoli, M., AkbariTorkestani, J., & Nazari, S. (2024). An optimization method in wireless sensor network routing and IoT with water strider algorithm and ant colony optimization algorithm. *Evolutionary Intelligence*, *17*(3), 1527-1545.
70. Krishnan, V. G., Saleem, A. P., Kirubakaran, N., Veeramalai, S., Kumar, A. K., Jehan, C., ... & Dhanalakshmi, G. (2023). Ensemble Deep Learning Classifier with Optimized Cluster Head Selection for NIDS in MANET. *J. Inf. Sci. Eng.*, *39*(6), 1233-1246.
71. Madhavi, K. R., Nawi, M. N. M., Reddy, B. B., Baboji, K., Kishore, K. H., & Manikanthan, S. V. (2023). Energy efficient target tracking in wireless sensor network using PF-SVM (particle filter-support vector machine) technique. *Measurement: Sensors*, *26*, 100667.

72. Elsadig, M. A. (2023). Detection of Denial-of-Service Attack in Wireless Sensor Networks: A lightweight Machine Learning Approach. *IEEE Access*.
73. Kumar, S., Kumar, S., & Garg, R. (2023). Range-free localization for GWSN using k-NN algorithm with local linear Gaussian kernel regression (KGR). *Evolving Systems*, *14*(1), 85-100.
74. Madasamy, N. S., Eldho, K. J., Senthilnathan, T., & Deny, J. (2024). A Novel Back-Propagation Neural Network for Intelligent Cyber-Physical Systems for Wireless Communications. *IETE Journal of Research*, *70*(2), 1361-1373.
75. Cui, H., Wang, S., & Zhou, C. (2023). A high-accuracy and low-energy range-free localization algorithm for wireless sensor networks. *EURASIP Journal on Wireless Communications and Networking*, *2023*(1), 37.
76. Adday, G. H., Subramaniam, S. K., Zukarnain, Z. A., & Samian, N. (2024). Investigating and Analyzing Simulation Tools of Wireless Sensor Networks: A Comprehensive Survey. *IEEE Access*.
77. Liazid, H., Lehsaini, M., & Liazid, A. (2023). Data transmission reduction using prediction and aggregation techniques in IoT-based wireless sensor networks. *Journal of Network and Computer Applications*, *211*, 103556.
78. Muruganandam, S., Joshi, R., Suresh, P., Balakrishna, N., Kishore, K. H., & Manikanthan, S. V. (2023). A deep learning based feed forward artificial neural network to predict the K-barriers for intrusion detection using a wireless sensor network. *Measurement: Sensors*, *25*, 100613.
79. 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. *Energies*, *16*(3), 1385.
80. Asha, A., Arunachalam, R., Poonguzhali, I., Urooj, S., & Alelyani, S. (2023). Optimized RNN-based performance prediction of IoT and WSN-oriented smart city application using improved honey badger algorithm. *Measurement*, *210*, 112505.
81. Verma, C., & Gupta, C. P. (2024, March). Analysis of Impact of The Charging on Propogation Dynamics of Malware in Rechargeable Wireless Sensor Networks. In *2024 2nd International Conference on Disruptive Technologies (ICDT)* (pp. 1184-

1190). IEEE.

82. Subramani, S., & Selvi, M. (2023). Intelligent IDS in wireless sensor networks using deep fuzzy convolutional neural network. *Neural Computing and Applications*, *35*(20), 15201-15220.
83. Noohi, M., Faraji, A., Sadrossadat, S. A., Mirvakili, A., & Moftakharzadeh, A. (2023). Modeling and implementation of a novel active voltage balancing circuit using deep recurrent neural network with dropout regularization. *International Journal of Circuit Theory and Applications*, *51*(5), 2351-2374.
84. Qi, Y., & Hossain, M. S. (2024). Harnessing federated generative learning for green and sustainable Internet of Things. *Journal of Network and Computer Applications*, *222*, 103812.
85. Chaturvedi, P., & Daniel, A. K. (2023). Neural Network Based Forecasting Technique for Wireless Sensor Networks. *Neural Processing Letters*, *55*(1), 671-687.
86. Balachander, T., Ramana, K., Mohana, R. M., Srivastava, G., & Gadekallu, T. R. (2023). Cooperative spectrum sensing deployment for cognitive radio networks for Internet of Things 5G wireless communication. *Tsinghua Science and Technology*, *29*(3), 698-720.
87. Zhang, G., Li, Y., Ji, B., Chen, K., Mu, Y., & Wang, W. (2024). M-ary Distributed Decision Fusion for Multi-Hop Relay Wireless Sensor Networks: Decision Fusion Rule, Implementation Framework, and Performance Analysis. *IEEE Internet of Things Journal*.
88. Baazaoui, M. K., Ketata, I., Fakhfakh, A., & Derbel, F. (2023). Modeling of Packet Error Rate Distribution Based on Received Signal Strength Indications in OMNeT++ for Wake-Up Receivers. *Sensors*, *23*(5), 2394.
89. Tabatabaei, S. (2024). A Fault-Tolerant Clustering Approach for Target Tracking in Wireless Sensor Networks. *Wireless Personal Communications*, 1-20.
90. Tagare, T. S., & Narendra, R. (2023). Confidentiality Enhancement Using Spread Spectrum Modulation Technique for Aggregated Data in Wireless Sensor Networks. *J. Commun.*, *18*(5), 294-300.
91. Kavra, R., Gupta, A., & Kansal, S. (2023). Optimization of energy and delay on interval data-based graph model of wireless sensor networks. *Wireless Networks*, *29*(5),

2293-2311.

Publications

Moka, M., Serome, K. & Selvaraj, R. An Iterative Convolutional Neural Network Based Malicious Node Detection (ICNNMND) Protocol For Internet of Things. Wireless Pers Commun 137, 1217-1232 (2024). https://doi.org/10.1007/s11277-024-11459-8

Wireless Personal Communications
https://doi.org/10.1007/s11277-024-11459-8

An Iterative Convolutional Neural Network Based Malicious Node Detection (ICNNMND) Protocol For Internet of Things

Moemedi Moka[1] · **Karabo Serome**[1] · **Rajalakshmi Selvaraj**[1]

Accepted: 30 June 2024
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

At present Internet of Things is widely used in various products and parts in our daily life. In the future, its uses and requirements will increase. However, there are several issues with the IoT network, one of which is secure data transmission. Also, malicious nodes in the IoT network are very difficult to identify accurately and quickly, whereas it is very important to do so. In the current context, several solutions are being proposed as there is a need to focus on setting up secure IoT frameworks, and as well as the protocols that runs on them. This research proposes a secure protocol called iterative Convolutional Neural Network based Malicious Node Detection protocol that uses the erroneous assumption learning problematic technique. This protocol will make changes in the application layer, the network layer and the physical layer in order to achieve secure data transmission in IoT. The proposed ICNNMND protocol will be able to detect attacks and eliminate them hence providing maximum data accuracy in IoT. For this, machine learning method will be used and the changes in the characteristics of the malicious nodes will be learned and their boundaries be explored. Then its error variation will be calculated and the malicious nodes be differentiated.

Keywords Convolutional neural network · Malicious node detection · Internet of Things · Protocol

Karabo Serome and Rajalakshmi Selvaraj these authors contributed equally to this work.

✉ Moemedi Moka
moemedi.moka@studentmail.biust.ac.bw

Karabo Serome
karabo.serome@studentmail.biust.ac.bw

Rajalakshmi Selvaraj
selvarajr@biust.ac.bw

[1] Department of Computer Science and Information Systems, Botswana International University of Science and Technology (BIUST), Palapye, Central District, Botswana

Published online: 11 July 2024