Deanship of Graduate Studies
Al-Quds University

Developing an Energy-Efficient Hybrid Intrusion Detection System for Wireless Sensor Networks

Murad Mohammad Fuad Jamal

M.Sc. Thesis

Jerusalem – Palestine
1447/هـ2025

Developing an Energy-Efficient Hybrid Intrusion Detection System for Wireless Sensor Networks

Prepared By:

Murad Mohammad Fuad Jamal

Supervisor: Dr. Raid Zaghal

This Thesis submitted in partial fulfillment of requirements for the degree of Master of Computer Science, Faculty of Graduate Studies—Al-Quds University

1447/هـ 2025م

Al-Quds University
Deanship of Graduate Studies
Computer Science

Thesis Approval

Developing an Energy-Efficient Hybrid Intrusion Detection System for Wireless Sensor Networks

Prepared by: **Murad Mohammad Fuad Jamal**

Registration No: **22220089**

Supervisor: **Dr. Raid Zaghal**

The Master's Thesis was submitted and accepted, Date: 20/12/2025

The names and signatures of the examining committee members are as follows:

1- Head of Committee: Dr. Raid Y. Zaghal, Signature: .....

2- Internal Examiner: Dr. Saeed A. Salah, Signature: .....

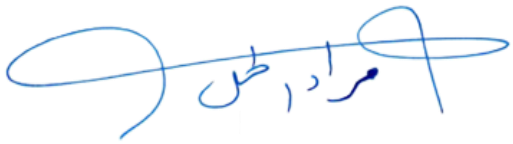
3- External Examiner: Dr. Muath N. Sabha, Signature: .....

Jerusalem – Palestine

1447/2025

Declaration

I certify that this thesis submitted for the degree of Master is the result of my own research, except where otherwise acknowledged, and that this study (or any part of the same) has not been submitted for a higher degree to any other university or institution.

Signed: 

Murad Mohammad Fuad Jamal

Date: 20 / 12 / 2025

Dedication

This work is lovingly dedicated to the memory of my dear father,

whose wisdom, kindness, and strength continue to guide me in every step I take.

Though you are no longer here, your spirit lives within my heart and achievements.

May your soul rest in eternal peace.

To my beloved mother,

whose unconditional love, prayers, and endless sacrifices shaped who I am today, your faith in me has been a light through every challenge, and your words of encouragement have carried me through the hardest days.

To my wonderful wife,

thank you for your patience, love, and unwavering support, you stood beside me in moments of exhaustion and doubt, reminding me that every effort has meaning when shared with those we love.

To my precious children,

you are the heart of my life and the reason I strive to be better each day.

May this achievement remind you that perseverance and faith can turn dreams into reality.

To my respected supervisor,

I am deeply grateful for your invaluable guidance, insightful advice, and kind encouragement throughout this research journey, your mentorship has left a lasting mark on both my work and my character.

Abstract

Wireless Sensor Networks (WSNs) are increasingly employed in various real-life applications, including smart cities, healthcare, and industrial systems; however, their limited computational resources make them more vulnerable to cyberattacks. Intrusion Detection Systems (IDS) that depend on Artificial Intelligence (AI) algorithms, including Deep Learning (DL) and Machine Learning (ML) algorithms, have shown their potential in detecting complex threat patterns. However, their high processing requirements restrict their deployment in WSN environments with constrained resources. This study solves the trade-off between detection accuracy and computational complexity. And it provides an energy-efficient deep learning model for intrusion detection in WSNs, in order to detect the latest cyber threats while maintaining high accuracy and low computational cost. We utilized the quantization optimization method over the Neural Network (NN) model to mitigate energy consumption. The findings of the study showed that the quantized model significantly reduces computational cost and memory usage while preserving good detection performance. The final prediction was made by employing an ensemble approach and combining all three models: DT, SVM, and the quantized NN. The final results showed that the ensemble model achieved an accuracy of 0.9238 with a prediction time of 4.1391 seconds, demonstrating a strong balance between detection effectiveness and computational efficiency. In addition, the ensemble attained high precision (0.9965), recall (0.9232), and F1-score (0.9585), indicating reliable detection of both normal and malicious traffic. Compared with individual models and evaluations on the NSL-KDD dataset, the proposed approach exhibited improved robustness and generalization while maintaining a compact memory footprint suitable for resource-constrained WSN environments.

Keywords: Artificial Intelligence (AI), Deep Learning (DL), Machine Learning (ML), Wireless Sensor Networks (WSNs), Intrusion Detection System (IDS), Quantization.

List of Figures

List of Tables

Table of Contents

Chapter One

Introduction

1.1 Overview

Wireless Sensor Networks (WSNs) provide a cost-effective way to collect and transmit sensing data across a wide range of applications, including environmental monitoring, industrial automation, and critical infrastructure. WSN nodes are typically resource-constrained; they have limited CPU capability, memory, and battery energy. These constraints make it difficult to apply conventional security mechanisms designed for richer computing platforms. At the same time, the distributed and wireless nature of WSNs exposes them to a variety of cyber threats, so effective intrusion detection mechanisms are essential to preserve confidentiality, integrity, and availability.

Intrusion Detection Systems (IDSs) monitor network activity and raise alerts for suspicious behavior. Recently, Machine Learning (ML) and Deep Learning (DL) methods have improved detection accuracy by learning complex traffic patterns and anomalies. However, state-of-the-art DL models are often large and compute-intensive, which challenges their deployment on low-power WSN nodes. This thesis investigates whether combining lightweight ML classifiers with quantized DL components can yield high detection performance while meeting the strict energy and memory limits of WSNs.

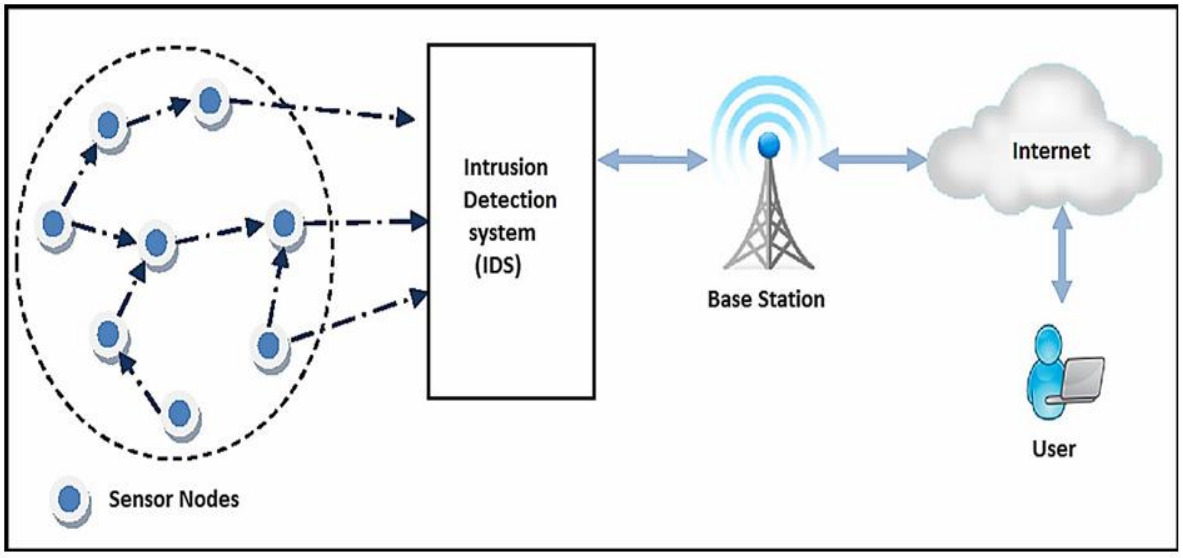


Figure 1.1 Protecting WSN using IDS (Gowdhaman & Dhanapal, 2022)

Figure 1.1 displays the architecture of the IDS approach in WSN. As seen, there are small devices on the left side of the illustration in a dashed circle distributed in the environment in order to gather and forward data; these devices communicate with each other, forming a WSN. IDS is located between the WSN and the base station; it mainly monitors the data coming from node traffic interactions. IDS analyzes the traffic, in order to detect intrusions/attacks. The base station is the central hub that gathers data from IDS. Mainly, it communicates with the external network and the IDS. This external network is the Internet, and the users are the base station. It grants access to WSN data in order to monitor the network, view alerts, and take corrective actions. The sensor nodes gather and forward data. IDS checks traffic abnormalities, and the base station sends secure data to the Internet to finally allow the user to remotely access information and intrusion alerts.

1.2 Problem Statement

WSN deployments increasingly demand automated, accurate intrusion detection, but existing ML/DL-based IDSs commonly assume abundant compute and energy resources. Large DL models achieve strong detection metrics on benchmark datasets but are impractical for many WSN scenarios because of high memory footprint, increased inference latency, and elevated energy consumption. Although model-compression techniques such as quantization exist, there is limited work that (i) integrates such optimizations into hybrid IDS architectures suitable for WSNs, and (ii) reports reproducible energy and resource estimates that relate directly to deployment trade-

offs. As a result, a gap remains between high-accuracy IDS research and practical, energy-aware IDS design for resource-constrained WSNs (Khan et al., 2020).

1.3 Research Questions

This study addresses the following research questions focused on balancing detection performance with resource efficiency in WSN environments:

1. How can ML and DL models be combined and adapted to detect intrusions efficiently in WSNs?
2. What specific computational, memory, and energy constraints most significantly affect the deployment of intrusion detection systems on WSN nodes?
3. What is the impact of applying quantization on detection performance (accuracy, precision, recall) and on model size?
4. To what extent can quantization and hybrid IDS design reduce memory usage, computational cost, and energy consumption while still meeting WSN latency constraints and maintaining reliable detection accuracy?

1.4 Research Objectives

To answer the questions above, this thesis pursues the following objectives:

1. Design an energy-efficient, quantization-aware intrusion detection framework for Wireless Sensor Networks, explicitly considering computational, memory, and power constraints.
2. Develop a hybrid IDS architecture that combines lightweight machine learning models (Decision Tree and Support Vector Machine) with a compact, post-training quantized deep learning model to achieve high detection accuracy with reduced resource consumption.
3. Apply feature engineering and selection techniques to construct a lightweight and informative feature set suitable for low-power WSN nodes, thereby minimizing preprocessing overhead.
4. Evaluate the impact of quantization and ensemble learning on detection performance and efficiency.

1.5 Research Hypothesis

The study tests the following hypotheses:

- **H1:** ML and DL models can effectively identify intrusions in WSN traffic; hybrid designs can outperform standalone traditional models.
- **H2:** Quantization significantly reduces model memory footprint and computational cost, enabling deployment on constrained WSN hardware.
- **H3:** Quantized models preserve acceptable detection performance (accuracy, precision, recall, F1-score), with only modest degradation relative to full-precision models.
- **H4:** DL models yield larger relative efficiency gains from quantization than classical ML models because of their higher baseline complexity.
- **H5:** A quantization-optimized hybrid IDS can achieve near-real-time inference while respecting typical WSN resource limits

1.6 Research Scope

This thesis focuses on the design, implementation, and evaluation of a hybrid, quantization-aware IDS for WSNs. Evaluation is performed using benchmark intrusion datasets (for example, NSL-KDD and UNSW-NB15) and simulation-based experiments that measure detection performance, model size, inference time, and estimated energy per inference. The primary optimization technique examined is quantization; other compression methods (e.g., pruning, knowledge distillation, low-rank factorization) are out of scope. A key limitation is the absence of a full hardware deployment: energy values reported are produced by analytical estimation and software-level timing measurements. Where appropriate, the thesis identifies how hardware measurements could validate and extend these results.

1.7 Research Contributions

This research provides a unique contribution when compared to the existing literature; the following contributions distinguish our research from current IDS studies for WSNs:

1. It uses two datasets to enable generalizability and robustness over different types of traffic. While most of the previous literature employed only one dataset.

2. It explicitly designed a new feasible approach for constrained WSN environments that measures energy, memory and computational efficiency, demonstrating the real feasibility of sensor nodes.
3. Also, it proposed a lightweight ensemble mechanism that combines several models, including Support Vector Machines (SVM), Decision Trees (DT), and Dense Neural Network (DNN), followed by a quantization approach. This integration is novel because it provides a balance between robustness to high-dimensional data, interpretability, and representation learning while remaining optimized for WSN networks. Existing literature often relies deep learning, which are computationally expensive for sensor nodes.
4. To attain the best possible trade-off between performance and resource consumption, model parameters were fine-tuned (e.g., DT with balanced weights and regulated depth, SVM with RBF kernel and tuned, and DNN with optimizer and layers).
5. This research highlights the trade-off between accuracy and resource consumption, in contrast to much IDS research that ends with accuracy reporting. It demonstrates that the suggested quantized ensemble IDS can operate on edge-level WSN nodes in a practical manner. This innovation guarantees that the work is both deployment-ready and theoretically sound for mission-critical WSN applications.

1.8 Overview of the Proposed Methodology

This research follows a systematic methodology to develop and evaluate an energy-efficient hybrid intrusion detection system for Wireless Sensor Networks:

- Benchmark intrusion datasets are preprocessed through feature encoding, normalization, class balancing, and feature selection to obtain lightweight representations suitable for constrained environments.
- Multiple detection models are trained, optimized, and compressed using post-training quantization.
- The optimized models are integrated into a heterogeneous ensemble and evaluated using detection performance metrics alongside resource-oriented measures such as memory usage, inference time, and estimated energy consumption.

1.9 Research Structure

The thesis is organized as follows. Chapter 2 reviews background and related work on IDSs, ML/DL for intrusion detection, and model compression techniques for edge devices. Chapter 3 describes the proposed hybrid architecture, feature extraction, and the quantization methodology. Chapter 4 presents results, including detection metrics, resource and energy estimates, and a comparative analysis with baseline methods. Chapter 5 concludes with a summary of findings, limitations, and directions for future work.

Chapter Two

Literature Review

This chapter presents some of the literature review on developing IDS within WSN and a summary for the related studies in terms of purpose, dataset, and techniques.

2.1 Literature Review

The researcher reviews some of the literature review on developing IDS within WSN and a summary for the related studies in terms of purpose, dataset, and techniques.

2.1.1 IDS within WSN

Abduvaliyev et al., 2010 provided a hybrid mechanism to detect intrusions within WSNs. The main purpose of their work is to enhance energy efficiency and accuracy of detection by integrating multiple methods for detection, including anomaly detection methods and signature-based methods in cluster-based networks, where the WSN is structured in clusters and the head administers the communication and detection of intrusions over the network to mitigate overhead. The proposed anomaly detection method flags the abnormalities and misuse detection detects known signatures. They evaluated the proposed mechanism by simulation (particular network size, traffic patterns, or attack models). The results showed that the proposed IDS mechanism achieved a high detection rate, while reducing computational cost, which leads to better energy conservation within WSN.

Another study by Kannan & Srinivasan, 2023 developed an IDS for WSN (ZigBee) that balances energy efficiency and security. They implemented intensive monitoring of cluster heads to identify

abnormalities using ML models for anomaly detection in order to distinguish between normal and malicious traffic. The simulation of their methodology was conducted in NS2 network simulator using 200-500 nodes using various network topologies. They simulated different attack scenarios including flooding attacks, selective forwarding and black hole. Finally, they measured model performance using accuracy, False Positive Rate (FPR), network lifetime, and residual energy. The results showed that the model achieved high detection rate equal to 96% over various attacks. FPR was less than current anomaly detection mechanisms, and energy consumption was reduced by ~20–25**%**, prolonging WSN lifetime. Their methodology illustrated scalability with increased number of nodes.

Hierarchical WSNs—commonly deployed in remote environments—are susceptible to routing attacks such as Blackhole, Sinkhole, and Wormhole. Gebremariam et al., 2023 proposed an IDS approach to detect these types of attacks across hierarchical WSN networks, harnessing the power of ML techniques, particularly a hybrid mechanism that combines two ML models: Random Forest (RF) and XGBoost. They employed three benchmark datasets: NSL-KDD, UNSW-NB15 and CICIDS2017. The results showed that the proposed mechanism achieved near-perfect detection accuracy. Specifically, combining RF with XGBoost achieved 99.80% on KDD data, while on the other two datasets, the Cluster-Labeling K-Means (CLK-M) method achieved 100% classification accuracy. In general, the proposed system attained 99.46% accuracy score in several tasks over simulation including feature extraction, route discovery, and attack detection.

Sakthimohan et al., 2024 aimed to integrate energy-efficient routing and intrusion detection into a unified mechanism that can detect different types of attacks. The proposed mechanism is based on deep learning approaches, their system involves two phases to perform detection, the first phase includes clustering of sensor nodes using density-peak K-means, then pelican optimization was applied to choose the optimal head of clusters, the second phase involves using Principal Component Analysis (PCA) for dimensionality reduction, and Recurrent Neural Network (RNN) for classification with Smitsh activation function to introduce non-linearity to the network. They used the NSL-KDD dataset to evaluate the modeling phase, and simulations were conducted with up to 500 sensor nodes. The results showed that the energy consumption was reduced to 49.67 mJ. And the accuracy of intrusion detection was 99.76%, demonstrating that their mechanism optimized routing while minimizing communication and computation overhead.

The study by Talukder et al., 2024 used ML algorithms framework for intrusion detection within WSNs. The proposed model mitigates both overfitting and underfitting issues, reduces computational and memory demand, and also optimized the detection pipeline. They used a dataset that contained different types of intrusions related to WSNs. They performed some of the preprocessing operations over the dataset such as, feature standardization to ensure consistency and Synthetic Minority Over-sampling Technique (SMOTE) to generate synthetic minority samples, and Tomek link removal to clean overlapping examples, in order to produce a balanced dataset. Finally, they implemented Decision Trees (DT), Random Forests (RF), XGBoost and

Neural Networks to perform detection. The results showed that RF was the best model with 99.78% accuracy score on binary classification and 99.92% on multi-class classification.

Umar et al., 2025b utilized an energy efficient DL model to detect intrusions in edge computing environments, the main aim is to prioritizing energy efficiency, small model footprint, and real-time performance using DL model. They implemented two optimization techniques including quantization and knowledge distillation in order to balance accuracy of detections and energy consumption. The study benchmarks their mechanism against Convolutional Neural Network (CNN), RNN, Dense Neural Network (DNN), and hybrid CNN-LSTM. The results showed that their mechanism is compact, which is over 10× smaller than Dense NN or CNN models, with slightly lower in detection accuracy rate at 90%**.** The study illustrated how the implementation of AI-based IDS on limited hardware is made possible by aggressive model compression.

Guo et al., 2024 proposed a new approach that integrates DL models with binary quantization, in order to strike a balance between detection performance and communication overhead within WSNs. Each sensor in the network implements binary quantization, each measurement is converted to a single bit (0 or 1) before transmission. This significantly reduces the amount of data transmitted**,** lowering energy use and bandwidth requirements. The DL model is developed to extract relevant patterns and make near optimal decisions despite the limited input information. The sensors performed the quantization, and the fusion center used DL to analyze signals and make detection decisions. The results showed that they achieved near-optimal detection accuracy despite the aggressive data compression from binary quantization.

Karthikeyan et al., 2024 proposed an IDS that enhances the security for WSNs integrated with IoT, they employed NSL-KDD dataset. First, they performed some of the preprocessing operations including categorical feature encoding, cleaning and normalization, and feature scaling in order to ensure uniformity, then they applied Firefly Algorithm (FA) to select important features, in order to reduce dimensionality and improve the performance of the classifier. After that they implemented ML models to perform classification into normal and attack within WSN-IoT network. The classification that was performed is a binary classification, and the algorithms were chosen because of their high performance in high-dimensional spaces and ability to handle non-linear decision boundaries. These algorithms including Support Vector Machine (SVM), K-nearest Neighbor (KNN), and XGBoost. SVM model was trained and fine-tuned using Grey Wolf Optimizer (GWO) in order to efficiently searches the parameter space to minimize classification error, the findings showed that employing FA with SVM and GWO achieved significant accuracy with a 99.34% score. Then the KNN was also efficient in making predictions with Particle Swarm Optimization (PSO). And finally, the XGBoost was attained a 95.36% accuracy score in making predictions on unseen data.

Madhuri, 2022 proposed a novel IDS based on cluster level certificate revocation method (IDS-SCRT), particularly target node-level packet drop attacks in WSNs. They utilized reputation-based system, then they performed monitoring of neighbor behavior. When it is detected as malicious

node, its cryptographic keys are revoked. Hence; without valid keys, malicious nodes are isolated from the network, meaning that cannot participate in communication or routing. Additionally, their mechanism follows a secure and effective strategy to distribute the keys between nodes in order to ensure only trusted nodes maintain communication privileges. The system was simulated and tested using multiple scenarios**.** The results showed that the proposed mechanism was enhanced network security significantly by detecting malicious nodes and isolates them effectively. When their approach was compared to other works from the literature, it was outperformed traditional security algorithms used in WSNs. The key contribution of their work is presented by combining behavioral monitoring with cryptographic enforcement, which is more proactive than detection-only systems, so that they bridge the gap among detection and mitigation. The findings showed that the IDS-SCRT method was more secure, and more energy efficient as well as it was considered highly suitable for WSN environments vulnerable to node-level drop attacks.

Sundaramoorthy et al., 2024 performed enhancements of WSN security by developing a new IDS mechanism and handle challenges of cloud computing and WSNs including constraints of resources, scalability and evolving nature of attack and employed NSL-KDD dataset to detect intrusions. They developed a layered hybrid IDS model combining four robust methods including Semi-Supervised Iterative Refinement (SSIR) in order to perform feature selection for real time detection of cyber threats. This method helps in reducing errors due to data distribution shifts. Also, they utilized Long Short-Term Memory (LSTM) to learn temporal patterns and enhance sequential anomaly detection accuracy, and finally Multilayer Perceptron Neural Network (MLPNN) to do final classification. The proposed mechanism improved the WSN security and solved the mentioned limitations. The results showed that MLPNN achieved 99.9% detection accuracy, outperforming SSIR, OSVM, and LSTM individually.

2.1.2 IDS within IoT

Devi et al., 2025 proposed a new federated learning framework for lightweight intrusion detection approach within WSN environments. The framework mainly aimed to secure WSNs in smart cities against Distributed Denial-of-Service (DDoS) attacks while preserving privacy and energy. Their framework uses optimized models in order to enable real-time operation and preserve privacy in attack detection. They implemented various deep learning models including CNN and LSTM, to perform anomaly detection with least resources. They employed ToN-IoT and Smart-city/WSN datasets, including DDoS scenarios for implementation. The findings showed that the framework was energy efficient and highly accurate in detection.

Wang et al., 2023 utilized advanced DL models and optimized the deployment efficiency issues in terms of computation, storage and energy for resource-limited environment, aiming for real-world deployment. They implemented Bidirectional Long Short-Term Memory (BiLSTM) network in order to model temporal features and sequential features from IoT traffic. They also applied a pruning optimization mechanism to remove redundant parameters and reduce model size

and complexity. Additionally, they performed the dynamic quantization for model parameters and precision, adopting to more reduced footprint and computational load of IoT networks and devices dynamically. The findings showed that the model achieved high performance and remains efficient and lightweight.

Mustafa et al., 2025 aimed to provide a cross-layer IoT model that uses ML with lightweight cryptography in order to minimize resource waste while defending against attacks. They utilized ML and cryptography at various OSI layers, the main components are Role-Based Access Control (RBAC), layer specific ML model (temporal anomaly, validation and pattern learning), and adaptive lightweight cryptography**.** The first component provides access to WSN nodes, the second one detects temporal anomalies at the network layer. The last component, speck encryption dynamically adjusts cryptographic overhead depending on device state. The results showed that the prevention rate of unauthorized access was 95%, false positives were reduced by 32% compared to baseline IDS, which is significant. On the other hand, they achieved a 30% reduction in power consumption when using Speck vs. AES. This illustrates that lightweight cryptography combined with an ML model is feasible.

Aji Gautama Putrada et al., 2024 implemented quantization approach to reduce energy consumption and complexity in IDS within IoT networks and devices, and employed ML models to detect IoT botnets, including k-Nearest Neighbor (KNN), they incorporated random sampling approaches, 16-bit quantization and feature selection. Firstly, they performed simulations on botnet attacks, and also utilized Kaggle botnet data. The findings showed that the compression ratio reached 175×, outperforming approaches that use only feature selection or sampling. The 16-bit quantization did not modify feature value distributions, indicating that accuracy was not degraded.

Hernández et al., 2024 performed optimization to the Neural Network (NN) in order to handle the high computational cost of DL models. Particularly, they utilized CNN (e.g. MobileNet, ResNEt50, ResNet152, Xception, and VGG16) to perform detection of intrusions and optimized it using several quantization methods involving Post-Training Quantization (PTQ) and Quantization-Aware Training (QAT). These methods assist in optimizing energy efficiency of the model and its detection performance. PTQ includes applying the quantization after full training while QAT includes integrating quantization operations during training for better adaptation. They also performed the evaluation of NVIDIA's DL Accelerator (DLA 2.0**)** on the Jetson Orin to evaluate its impact on performance and energy efficiency. The main finding showed that, for on-device training feasibility, the Jetson AGX Orin can perform training autonomously, illustrating practical on-device model adaptation capability. In addition, the model performance on inference was significant, as PTQ provided better performance than QAT with faster and more accurate results, while QAT achieved higher energy efficiency. However, DLA accelerator results showed worse performance and efficiency when used with these models, possibly because of the architectural layer compatibility issues. Additionally, MobileNet-v2 was the best model with the lowest latency, as it benefited the most from the optimization process.

Sharmila & Nagapadma, 2023 performed optimization of autoencoder DL algorithm, the optimization was perfromed by combining clustering, pruning and integer quantization. They applied two variants of Quantized Autoencoder (QAE), the first variant (QAE-u8) utilized 8-bit quantization, and the QAE-Fp16 utilized 16-bit quantization. They trained the models on IoT traffic, RT-IoT2022 dataset was employed, during training, the reconstruction error was computed. The evaluation measurements included, accuracy**,** precision**,** recall**,** F1-Score, plus practical deployment metrics like memory usage**,** model size**,** and CPU utilization. The results showed that, the model surpassed other models in resource efficiency, achieving significant reduction in average memory consumption to 70.01%**,** also it achieved a significant compression in memory footprint with 92.23%. And finally, it achieved a reduction in peak CPU usage to 27.94%.

Acharya et al., 2023 presented a new mechanism that employs NN with quantization in order to provide a model with thousands of times smaller than a normal NN with high intrusion detection capability. They combine quantization-aware training with neural architecture search. The proposed approach allows for automatic discovery of NN topologies that are effective and also efficient in IDS within WSN. The optimized models were deployed on FPGA hardware, illustrating reduced latency and improved throughput, which are significant factors for real-time intrusion detection systems. The experimental results displayed that the quantized NN model achieved significant accuracy in data analysis for intrusion detection. Additionally, hardware testing results illustrating the practical applicability of their mechanism in real-world conditions. In their investigation, they provided a robust solution to the critical issue of implementing machine learning-based intrusion detection systems on hardware with constrained power, limited network connectivity, and limited processing capabilities.

Bella et al., 2025 presented a lightweight IDS for resource-constrained environments, such as IoT devices and WSNs. The main goal of their work was to make DL anomaly detection feasible on edge devices, this feasibility is reflected in low memory and computational overhead, without degradation in detector performance. They utilized QAE framework for anomaly detection, this framework includes applying pruning, clustering, and quantization. These techniques are implemented to an AE architecture. As well as the pruning can remove the redundant associations in network, while clustering mitigates the complexity of model by grouping similar weights. The main contribution is quantization in two variants 8-bit and 16-bit. The proposed framework employed RT-IoT2022 dataset to develop the model. This dataset includes real-time data of network traffic from different IoT applications. The autoencoder network was evaluated using various metrics such as accuracy, precision, recall, F1 score, as well as CPU utilization, memory usage, and model size. The results of their framework displayed that QAE-u8 model achieved a significant accuracy of 96–98%, precision of 98.39%, recall of 98.59%, and F1-score of 99.1%, while reducing model size by over 90% and lowering memory and CPU usage significantly.

Perumal et al., 2023 solved several challenges faced in IDS approaches for IoT systems such as long processing time, high latency, and inability to accurately detect intrusions. They proposed Vectorization-Based Boost Quantized Network **(**VBQ-Net). The main aims of the study include

reducing dimensions, perform efficient feature extraction, and develop a quantized NN classifier with employing Metaheuristic optimization. The architecture of VBQ-Net network, involves Vector Space Bag of Words (VSBW), in order to convert raw input data into a vector space representation, this step assists in reducing computational load for real-time IDS in resource-constrained IoT. Additionally, Boosted Variance Quantization Neural Network (BVQNN) was implemented in order to classify intrusion types. This model utilizes boosting and variance-based weighting of features in order to enhance classification accuracy. Furthermore, Multi-Hunting Reptile Search Optimization (MH-RSO) was also employed in their framework**,** by using a metaheuristic algorithm that helps in enhancing detection accuracy. This optimization technique computes probability scores in order to guide the classification process toward optimal intrusion prediction. Three datasets were employed in the study for evaluation, including IoTID-20, IoT-23, and CIDDS-001. These datasets entail various attacks types, such as DoS, DDoS, botnet**,** scanning, brute force, and more. The results showed that the proposed model was significant, and it achieved high detection accuracy across all datasets. They enhanced inference speed and reduced memory usage. The proposed mechanism was compared to traditional IDS and they found that it was outperformed them in terms of accuracy, precision**,** recall**,** and F1-score**.**

Ullah & Mahmoud, 2021 presented a new approach for intrusion detection within IoT networks, the approach depends on developing one of the most common deep learning algorithms. First, they developed three variants of CNN. The aim is to do classification into multiple classes. They used four datasets for evaluation including BoT-IoT, IoT-23, IoT Network Intrusion, and MQTT-IoT-IDS2020. They utilized transfer learning to implement both binary classification and multi-class classifications. The results showed that the models achieved significant performance when it is compared into current deep learning mechanisms. In multi-class classification, the CNN model attained 99.95% for all measurements including accuracy, precision, recall and f1-score. And for binary classification, it achieved a 99.96% accuracy, a 99.90% precision, a 99.95% recall, and a 99.93% f1-score.

Roy & Cheung, 2018 proposed a new mechanism to detect intrusions in IoT networks based on deep learning models, specifically they implemented Bi-directional Long Short-Term Memory (BLSTM) which belongs to RNN. The method performs binary classification of dataset samples into attack or normal based on sample features. They employed the publicly available dataset named UNSWNB15 IoT dataset for evaluation. First, the data was prepared to feed into the model by applying some preprocessing steps such as reducing the dataset, selecting features, and manual labeling. After preprocessing, the data was fed to the BILSTM model for classification. The experimental results showed that, the model was achieved an accuracy of 95%, which is significant.

Kim & Heo, 2022 performed meaningful feature extraction of the hydraulic system IoT sensor data, then they detected of the abnormal defects using ML and DL algorithms. They employed a dataset collected from IoT sensors and categorized into a set of clusters. To train the model, they selected 2335 features using the Boruta algorithm and correlation coefficients for each system

component. Several algorithms were implemented to perform the detection for system components including LightGBM, Linear Discriminant Analysis (LDA), logistic regression, Support Vector Classifier (SVC), DT, RF, XGBoost, and multi-layer perceptron. The evaluation measurements include True Positive Rate (TPR) and True Negative Rate (TNR). The results showed that internal pump leakage, valve condition, and hydraulic accumulator data achieved 0.94% TPR and 0.84% FNR.

J. R. Rose et al., 2021 performed anomaly detection for suspicious activities in IoT networks and devices. They combined network traffic profiling to establish behavioral baselines with machine learning to identify anomalies and classify traffic types. Testing was done using Cyber-Trust testbed tool, IoT network simulation with benign and malicious traffic. The findings showed that the proposed model was robust and delivered good performance with an accuracy of 98.35% and a FPR of only 0.98%.

The work done by Lima et al., 2024 provided IDS and focused on optimizing the proposed mechanism in preprocessing operations, rather than modeling phase and results. The authors performed evaluation of how data preprocessing techniques such as normalization, cleaning, feature selection and hyperparameter optimization impact the power of ML classifiers. And how they impact computational efficiency including training and testing times within intrusion detection systems. They employed two datasets (CSE-CIC-IDS2018 and KDD Cup 1999) and performed several preprocessing operations such as outlier removal using z-score, normalization using min-max scaler, feature selection using Pearson correlation. Furthermore, they utilized grid search combined with k-fold cross-validation for model fine-tuning to enhance predictive performance while avoiding overfitting. After that, they implemented various ML models including RF, DT, XGBoost, Naive Bayes, and Neural Network. The results showed that they achieved significant performance. The model that applied preprocessing and optimization of hyperparameters achieved higher performance in detecting intrusions, and faster training and testing time. In addition, the results showed that Random Forest, Decision Tree, and XGBoost benefited the most. Naive Bayes showed relatively weaker results, likely due to its simplistic assumptions.

Yagiz & Goktas, 2025 proposed a new mechanism for IDS in order to solve challenges faced in traditional IDS that includes limited explainability, high computational demands, and inflexibility against developed attack patterns. The main goal of the study is to provide a lightweight Explainable Network Security approach (LENS-XAI), this approach combines variational autoencoder models, knowledge distillation and attribution-based explainability methods in order to achieve high detection performance and transparency in making decisions. They employed a training set entails 10% of the available data, the proposed approach was optimized computational efficiency without affecting model performance. They employed 4 datasets —Edge-IIoTset, UKM-IDS20, CTU-13, and NSL-KDD to evaluate the proposed approach. The findings demonstrated that the approach was achieved very superior performance, with 95.34%, 99.92%, 98.42%, and 99.34%, detection accuracy, respectively. Moreover, they outperformed in reducing

FP and adapting to complex attack conditions, surpassing current state-of-the-art models. The strength point of LENS-XAI entails its lightweight design, its suitability for resource-constrained environments such as IoT and WSN, and its scalability over various cybersecurity contexts. In addition, the expandability application was improved transparency and trust which are very crucial in deployment. Their work contributes to develop IDS research by solving feature interpretability, computational efficiency, and real-world applicability.

Shen et al., 2024 presented a FLEKD framework, that integrates federated learning and ensemble knowledge distillation, in order to improve IDS in IoT networks by solving several challenges related to Non-Independent and Identically Distributed (non-IID) data, data privacy, and inefficiencies in model aggregation. The framework involved decentralized training, where each device trained model locally, and utilized probability (model output) from every local model to construct a global one using ensemble approach in order to mitigate heterogeneity and enhance generalization. They employed CICIDS2019 dataset for evaluation. The results showed that, the proposed framework was superior, as it enhanced detection of unknown/zero-day attacks. It was also achieved faster convergence and lower communication cost than traditional methods.

2.1.3 Synthesis and gaps

The surveyed literature reveals two dominant trends. First, many prior works focus on high-accuracy, compute-heavy DL models (CNNs, RNNs, LSTMs) that demonstrate excellent detection metrics on benchmark datasets but are not optimized for resource-constrained WSN deployments (e.g., Umar 2025, Acharya 2023, and Wang 2023). Second, a growing subset of research applies model-compression techniques — quantization, pruning, knowledge distillation, and neural-architecture search — to shrink models for edge devices, however, most quantization work evaluates only model size or accuracy; few studies report a rigorous energy-estimation methodology or compare end-to-end ensemble trade-offs (accuracy vs energy) in WSN contexts.

This thesis fills an important research gap in intrusion detection for WSN by focusing on both detection performance and deployment feasibility. Existing studies often prioritize accuracy using complex deep learning models while neglecting energy, memory, and latency constraints of WSN nodes. The proposed approach addresses this limitation through a lightweight heterogeneous ensemble that combines classical machine learning with a quantized deep neural network. Unlike prior work, the study provides compact, deployable model artifacts suitable for edge-level execution. It also presents reproducible estimates of energy consumption and memory usage. Together, these contributions demonstrate a practical and original IDS solution tailored for real-world WSN environments.

2.2 Summary

Table 2.1 summarizes the previous mentioned studies in terms of study objective, the utilized methods and the achieved results. Most prior IDS studies do not report energy consumption because they evaluate models in simulation environments or on general-purpose servers, where direct power measurement is unavailable or outside the study scope. Many works prioritize detection accuracy and algorithmic novelty, assuming energy efficiency implicitly from reduced model size or complexity rather than measuring it explicitly. In addition, there is no standardized methodology for reporting energy per inference, leading researchers to omit this metric to avoid unreliable or hardware-dependent claims.

Table 2.1 ,A , Comparative analysis of IDS approaches for WSN and IoT

Ref	Study Objective	Technique	Dataset	Result
(Abduvaliyev et al., 2010)	Detect intrusions within WSNs	Anomaly and misuse detection methods (SLIPPER). They utilized cluster-based wireless sensor networks (CWSNs) in order to reduce communication costs and packet overheads	They received data by simulation	high detection rate, while reducing computational cost, which leads to better energy conservation within WSN
(Kannan & Srinivasan, 2023)	IDS for WSN (ZigBee)	ML models	Simluation of attacks	High detection rate 96%, energy consumption lowered by ~20–25%,
(Gebremariam et al., 2023)	Propose an IDS approach to detect these types of attacks across hierarchical WSN networks	Two ML models which are Random Forest (RF) and XGBoost.	NSL-KDD, UNSW-NB15 CICIDS2017	99.46% detection accuracy
(Sakthimohan et al., 2024)	Unify energy efficient routing and intrusion detection into a single mechanism	Density-peak k-means, pelican optimization, PCA, RNN	NSL-KDD dataset	Energy consumption was reduced to 49.67 mJ. And the accuracy of intrusion detection equals 99.76%
(Talukder et al., 2024)	Provide an Intrusion detection model within WSN	Decision Trees (DT), Random Forests (RF), XGBoost and Neural Networks	(WSN DS) - 374,661 records (Blackhole, Flooding, Grayhole, and Scheduling)	RF (Best) 99.78% accuracy (binary) 99.92% accuracy (multi-Class)

Table 2.2 ,B , Comparative analysis of IDS approaches for WSN and IoT

(Umar et al., 2025b)	Detect intrusion in edge computing environments in order to prioritize energy efficiency by aggressive compression.	Deep learning models (CNN, RNN, DNN)	CICIDS2017 dataset	Accuracy rate at 90%. with (~20 KB)
(Guo et al., 2024a)	Integrates DL models with binary quantization	DNN to train the probability controller in the quantizer separately to the detector at the FC.	generated via Gaussian observation	They achieved near-optimal detection accuracy with reduced complexity due to the compression and binary quantization
(Karthikeyan et al., 2024)	Develop an IDS within WSN and IoT to enhance security	SVM/KNN/ XGBOOST – classification GWO – parameters tuning FA –feature selection	NSL-KDD Dataset	FA-ML (SVM with FA + GWO): 99.34% accuracy KNN-PSO: 96.42% XGBoost: 95.36%
(Madhuri, 2022)	novel IDS within WSN	cluster level certificate revocation method (IDS-SCRT) method	Simulation	Energy consumption ~6.5 J Routing overhead~0.02%
(Sundaramoorthy et al., 2024)	Enhance security of WSN and cloud environments	SSIR, OSVM, and LSTM and MLPNN	NSL-KDD	99.9 detection accuracy of MLPNN
(Devi et al., 2025)	Secure WSNs in smart cities by providing lightweight intrusion detection approach	applying Federated Learning framework, CNN, LSTM	ToN -IoT data	The framework was energy efficient and highly accurate in detection.
(Wang et al., 2023a)	Optimize the deployment efficiency issues	BiLSTM and pruning optimization mechanism	CIC IDS2017, N-BaIoT, CICIoT202	The model was achieved high performance and remains efficient and lightweight.
(Mustafa et al., 2025)	Provide a cross-layer IoT model and minimize resource waste	cryptography at various OSI layers, (RBAC), temporal anomaly using ML model	They collected simulation logs it includes, SNR, routing intervals, payload entropy, and temporal access logs	prevention rate = 95%, FP =32% and 30% reduction in power consumption

Table 2.3 ,C , Comparative analysis of IDS approaches for WSN and IoT

(Aji Gautama Putrada et al., 2024)	Provide IDS within IoT networks and devices and perform quantization	KNN	botnet data	16-bit quantization Doesn't modify feature value distributions and accuracy was not degraded
(Hernández et al., 2024)	Handle cost challenge in IoT network by PTQ and QAT	MobileNet, ResNEt50, ResNet152, Xception, and VGG16	Flickr-Faces-HQ (FFHQ) dataset.	Superior energy efficiency and high precision
(Sharmila & Nagapadma, 2023a)	Optimize consumption efficiency of IoT IDS using quantization	Autoencoder DL algorithm and quantization	RT-IoT2022 dataset	Reduced memory consumption 70.01%, CPU usage 27.94%.
(Acharya et al., 2023)	Design efficient NN for IDS for hardware deployment in constrained IoT devices	NN with quantization-aware NN	Not specified	The resulting network is ~1,000× smaller than state-of-the-art and uses 2.3× to 8.5× fewer LUTs on FPGA while retaining comparable performance
(Bella et al., 2025)	Targets resource-constrained IoT/edge environments, similar to WSN scenarios	Incorporates pruning, clustering, and integer quantization within an autoencoder-based IDS	RT-IoT2022	98.40% accuracy, while reducing model size by over 90% and lowering memory and CPU usage significantly.
(Perumal et al., 2023)	Improve cybersecurity in (IoT) environments.	Vector Space Bag of Words (VSBW) , Boosted Variance Quantization Neural Network (BVQNN), Multi-Hunting Reptile Search Optimization (MH-RSO)	IoTID-20, IoT-23 , CIDDS-001	- High detection accuracy across all datasets, - Because of the quantization and vectorization, they improved inference speed and reduced memory usage
(Ullah & Mahmoud, 2021)	Proposed an IDS based on DL models	Convolutional Neural Network	BoT-IoT, IoT-23, IoT Network Intrusion, and MQTT-IoT-IDS2020	Accuracy = 99.95 (binary classification) Accuracy = 99.96 (multiclass classification)

Table 2.4 ,D , Comparative analysis of IDS approaches for WSN and IoT

(Roy & Cheung, 2018)	Propose an IDS based on Recurrent Neural Network DL models	Bi-directional Long Short-Term Memory (BLSTM)	UNSW-NB15	Accuracy = 95%
(Kim & Heo, 2022)	Anomaly Detection with Feature Extraction Based on Machine Learning	(LDA), LightGBM, (LDA), LR , (SVC), DT, RF, XGBoost, and MLP	Collected data from IoT sensors	0.94 True Positive Rate and 0.84% False Negative Rate.
(J. R. Rose et al., 2021)	Anomaly Detection from IoT networks	ML model	generated using the Cyber-Trust testbed	Accuracy of 98.35% and a false-positive rate of 0.98%.
(Lima et al., 2024)	Optimize IDS through significant preprocessing	ML models including (DT, RF, XGBoost, NB)	CSE-CIC-IDS2018, KDD Cup 1999	Higher detection rate Better training and testing time
(Yagiz & Goktas, 2025)	Optimize IDS limitations of feature interpretability, computational efficiency, and real-world applicability.	this approach combines variational autoencoder models, knowledge distillation and attribution-based explainability methods,	Edge-IIoTset, UKM-IDS20, CTU-13, and NSL-KDD	95.34%, 99.92%, 98.42%, and 99.34% detection accuracy
(Shen et al., 2024)	improve IDS in IoT networks	integrates federated learning and ensemble knowledge distillation	CICIDS2019	- improve detection of unknown/zero-day attacks. - Faster convergence and lower communication cost than traditional methods

Chapter 3

Methodology

This chapter explains the methodology of the study, illustrating the employed dataset, the utilized methods and mechanisms, evaluation measurements and the flowchart of the proposed methodology. It also provides a detailed explanation of the implemented workflow, illustrating how the results and the initial objectives will be archived.

3.1 Dataset

This study utilized two benchmark datasets to implement the proposed methodology including NSL-KDD dataset and UNSW-NB15. This section explains the characteristics of these two datasets as follows:

- **NSL-KDD dataset**

Network Security Laboratory – Knowledge Discovery and Data Mining (NSL-KDD) data was mainly collected and designed in order to evaluate IDS systems in the networks. It is an enhanced version of the original KDD'99 dataset, to handle its issues such as unbalancing and duplicate records. The data is labeled as normal or attack, and the attack has four types (DoS, Probe, R2L, U2R). The dataset has 125,973 records in the training set and 22,544 records in the testing set. It also includes 41 features for each record. These features entail basic features like protocol types, content features like number of failed login attempts and traffic features (Tavallaee et al., 2009a).

- **UNSW-NB15**

UNSW-NB15 (University of New South Wales Network-Based 2015) the dataset involves real network traffic with a mix of normal and malicious activities was collected to provide a comprehensive dataset for IDS evaluation. It involves ~2.5 million raw network packets, these packets were combined into 257,673 flows for ML experiments. It has 49 features including flow features, basic features, content features, time-based features, and some additional features like state of connection. The data is labeled as normal or attacks, attacks records include 9 types of attacks: fuzzers, analysis, backdoor, DoS, exploits, generic, reconnaissance, shellcode, and worms (Moustafa & Slay, 2016).

Table 3.1 provides a comparative overview of the both datasets used in this study, highlighting their structural characteristics, feature composition, and attack diversity.

Table 3.1 Comparative overview of NSL-KDD and UNSW-NB15 datasets used for intrusion detection evaluation

	NSL-KDD	**UNSW-NB15**
Dataset Origin	Enhanced version of KDD'99	University of New South Wales
Data Type	Network connection records	Real network traffic flows
Total Samples	148,517	257,673
Training Samples	125,973	175,341
Testing Samples	22,544	82,332
Number of Features	41	49
Feature Categories	Basic, content, traffic-based	Flow-based, basic, content, time-based
Label Type	Binary (Normal / Attack)	Binary (Normal / Attack)
Attack Categories	4 (DoS, Probe, R2L, U2R)	9 (Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms)
Class Imbalance	Moderate	High (e.g., Worms underrepresented)
Traffic Realism	Simulated	Realistic modern traffic

3.2 Data Preprocessing

Preprocessing involves preparing a dataset for modeling. Each dataset has its own specific processing steps based on its characteristics and content. This section presents the preprocessing processes performed on the two datasets. Table 3.2 illustrates the conducted preprocessing operations over UNSW-NB15 dataset. Irrelevant identifiers were removed to prevent bias and reduce noise, while one-hot encoding was used to correctly represent categorical network attributes without imposing artificial ordering. Min–Max normalization was chosen to place all features on a comparable scale, improving model convergence and ensemble integration. SMOTE was employed to address class imbalance, enabling the models to learn minority attack patterns more effectively and improving overall detection reliability.

Table 3.2 Preprocessing steps applied to the UNSW-NB15 dataset

Preprocessing	Description
Remove Irrelevant Features	Drop unimportant features for classification (e.g., id, attack_cat)
Encode Categorical Features	Proto, state and service features were converted into numerical format using one-hot encoding.
Data Normalization	Data features were scaled using Min–Max Scaling.
Handle Class Imbalance	In UNSW-NB15, some attack categories (like Worms) are underrepresented. Solution: SMOTE oversampling
Split Dataset	Dataset was divided into training set and testing set, 175341 records for training and 82332 for testing.

Table 3.3 shows the conducted preprocessing operations over NSL-KDD data. Binary label conversion was applied to align the dataset with the binary intrusion detection objective of this study. Categorical encoding and feature scaling were necessary to transform heterogeneous feature types into a uniform numerical representation suitable for ML and DL models. Dimensionality reduction via PCA was used to mitigate feature redundancy and reduce computational complexity, followed by RFE to retain the most informative features. This combination ensures a compact, relevant feature set that balances detection performance with the resource constraints typical of WSN environments.

Table 3.3 Preprocessing steps applied to the NSL-KDD dataset

Preprocessing	Description
Binary Label Conversion	Labels were converted into 0 (normal) and 1 (malicious)
Drop unnecessary columns	'Difficulty' column was removed
Encode Categorical Features	One-hot encoding was utilized to encode the following categorical features ['protocol_type', 'service', 'flag']
Feature Scaling	Min–Max Scaling
Dimensionality Reduction	Principal Component Analysis (PCA) was applied to reduce dimensionality, reduced from 122 to 50 components
Feature Selection	Recursive Feature Elimination (RFE) was applied to feature selection - reduced features to 43
Split features and labels	Data was divided into 125973 samples for training and 22544 for testing

3.3 Methods

This section discusses the methods utilized in this study, including DT, SVM and Dense Neural Network (DNN).

- **Decision Tree (DT)**

This algorithm is a supervised ML model; it is widely utilized for classification and regression tasks. DT models' decisions and their possible outcomes in a tree structure as shown in Figure 3.1. The nodes represent a feature or attribute, branches represent decision rules, and leaves represent the output or class label (for classification) (Costa & Pedreira, 2023). The operation of DT involves several steps; the first step includes selecting best features for splitting based on some criteria (e.g. Information gain). Then the data is split into subsets based on values of the selected feature. The subset is further split using the same process until no features remain to split, and all samples belong to the same class. Finally, when stopping criteria are met, the leaf node is assigned a class label.

There are many advantages of DT include that it is easy to interpret, can handle categorical and numerical data, and can capture non-linear associations among features. On the other hand, there are many disadvantages, such as it is prone to overfitting if the tree grows too deep, and it may be

unstable, meaning that small changes in data may produce a very different tree (Charbuty & Abdulazeez, 2021).

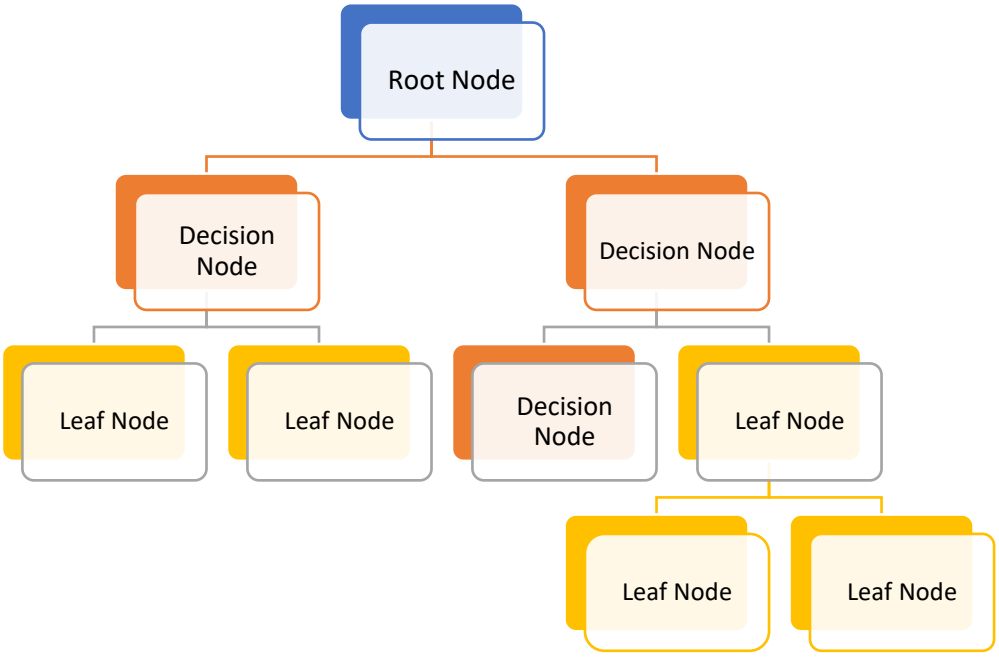


Figure 3.1DT structure (Charbuty & Abdulazeez, 2021)

Figure 3.2 shows a classic example of a decision tree diagram from the literature (Staudemeyer, 2012).

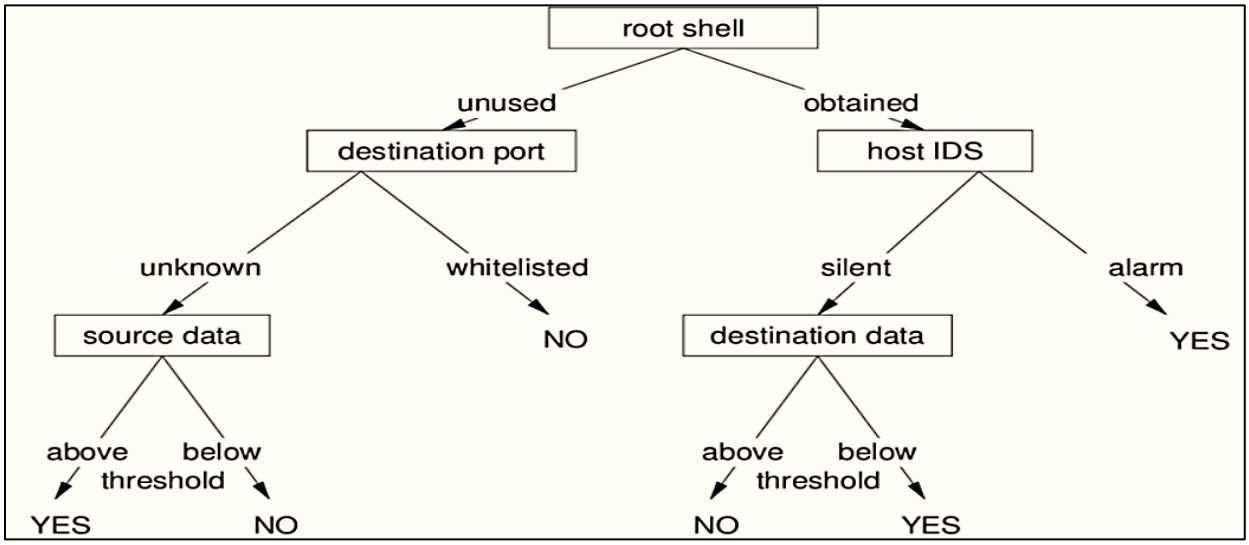


Figure 3.2 Example of DT (Staudemeyer, 2012)

The figure depicts the following:

- **Root Node**: It is the initial feature or initial condition, such as "authentication used".
- **Branches:** Branches represent conditions, and based on the value of this condition, the flow moves through different branches for example, whether authentication is valid or invalid).
- **Internal Splits**: These splits represent additional decisions that are made—for example, checking properties like "shared paths allowed?".
- **Leaf Nodes**: These nodes represent final nodes (classification outcome), YES indicates that an intrusion (U2R attack) is detected and NO indicates benign behavior (no attack).

Every path beginning from root to leaf indicates to a logical rule utilized in order to classify the behavior as normal or intrusive.

In this work, decision tree was selected due to its efficiency in training and prediction, also because of its interpretability advantage over other models, as it clearly representing decision rules, which makes it a good candidate for WSNs.

- **Support Vector Machine (SVM*)***

SVM is also a supervised ML model, it is widely utilized for classification, but it can also handle regression tasks. The main idea of this algorithm is to find the optimal boundary (hyperplane) that separates classes in the feature space. For linear data, the hyperplane perfectly divides the classes. For non-linear data, the algorithm uses kernel functions to map data into a higher-dimensional space where it becomes linearly separable. First, it finds optimal hyperplane, then it uses kernel tricks to handle non-linear data, by performing transformations using polynomial kernel, Gaussian or sigmoid kernel. It then solves the optimization problem by ensuring maximum margin and correct classification (Shmilovici, 2005).

SVM has many advantages including, working well with high dimensional data, and it is powerful when number of features greater than number of samples. As well as, it is also very robust to overfitting, especially when properly tuned with kernels and regularization. However, for large datasets, SVM is very computationally expensive, and it is less interpretable in comparison with DT. Additionally, choosing the right kernel and parameters is critical. In WSN context, SVM can classify network traffic into attack or normal. It works well with high-dimensional data from nodes, using the Radial Basis Function (RBF) kernel to capture complex non-linear relationships to detect intrusions (Shmilovici, 2005). Figure 3.3 shows an example of SVM (Ahmed et al., 2025) and displays an intuitive grasp of how SVM separates classes and the importance of margin maximization.

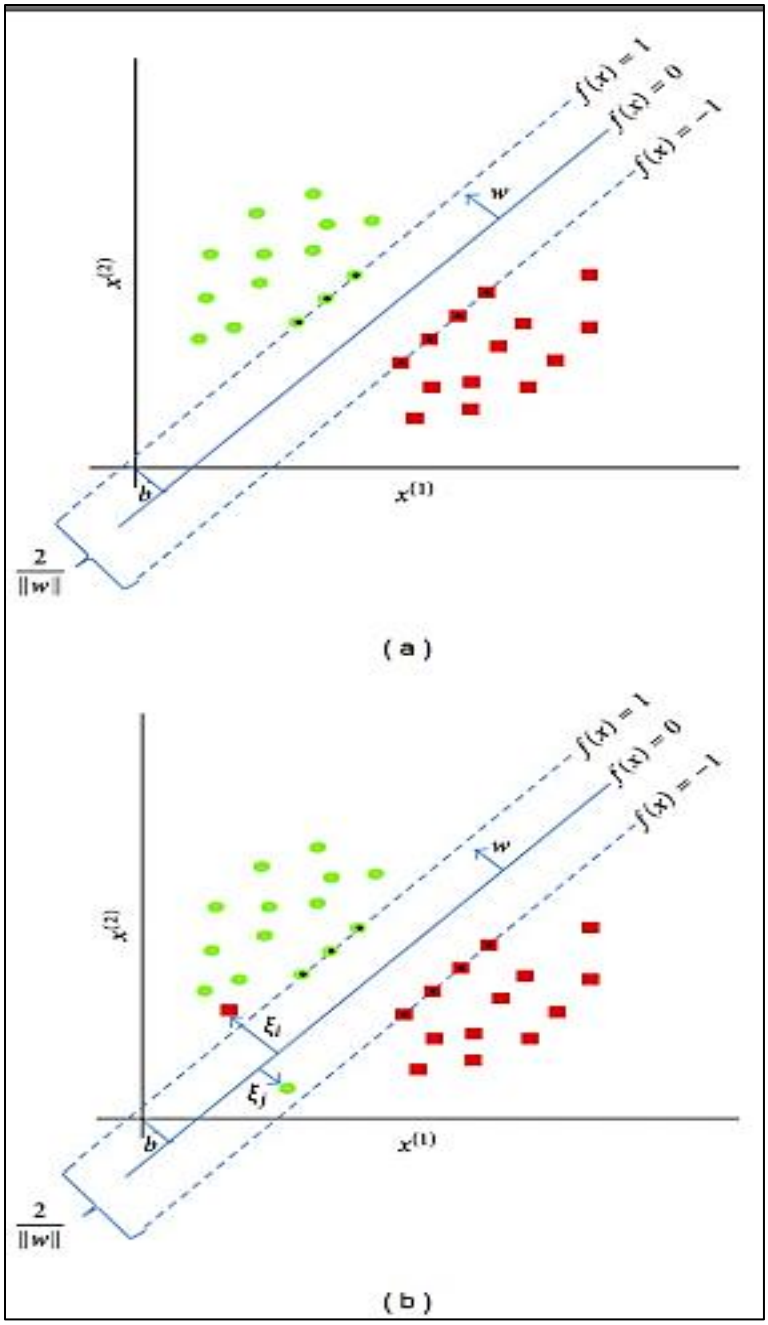


Figure 3.3 SVM example (Ahmed et al., 2025)

The figure 3.3 illustrates a 2D feature space, the red points represent the anomaly sample of traffic data and the green points represent normal samples of traffic data. The solid line in the figure displays the separation of two classes (normal and anomaly), which is the SVM boundary of decision called (hyperplane). The shaded regions indicate the model's predicted classification area, and the margin describes the gap between two dashed boundary lines for 2 classes marks the maximal separation the SVM achieves between the closest points (support vectors) of each class.

This figure shows how SVM strives to maximize the margin and achieve robust classification between normal and anomalous network behavior.

In this work, the SVM algorithm was selected due to its efficiency in handling high dimensionality feature spaces, which are common in intrusion detection, such as UNSW-NB15 and NSL-KDD. Additionally, SVM model utilize different kernel functions in order to capture non-linear relationships among features from tabular data directly, in contrast to CNNs or LSTMs that need sequential or spatial patterns.

For the developed SVM model, we utilized the RBF kernel function to handle nonlinear associations. The kernel coefficient gamma regulated the effect of individual training points on the decision boundary. Compared to CNN or LSTM, SVM is lighter, does not require sequential dependencies, and works best in a WSN environment, where resource efficiency is critical (Kim et al., 2020).

- **Dense Neural Network (DNN)**

ANN provides a self-adaptive machine learning approach, powered by non-linear data, used for patterns classification. The biological neural network forming the human brain was inspiring for the reasoning behind ANN. In modeling, in cases where the relationship at hand is uncertain, ANN is very solid (Han et al., 2018).

An ANN consists of organized layers of neurons that are organized. The layers of neuron then transform the input into a specific output. A non-linear equation adds to it whenever a unit obtains an input. It moves to the next layer after this. Due to this sequence of events, as the input passes from one layer to the next until it transforms into an output, a neural network is a forward feed. No input is sent to the previous layer when an action occurs at a certain stage. Weights are added to the signals flowing from one device to another, which are tailored to a specific classification question in the training process by several modifying approaches such as back propagation approach (Rojas, 1996). ANNs can be trained using both supervised and unsupervised learning models, A key advantage is that the classifier is generalizable, meaning it can make predictions for inputs outside the training dataset. Figure 3.4 shows an example of Feed forward Neural Network (Vuckovic et al., 2002).

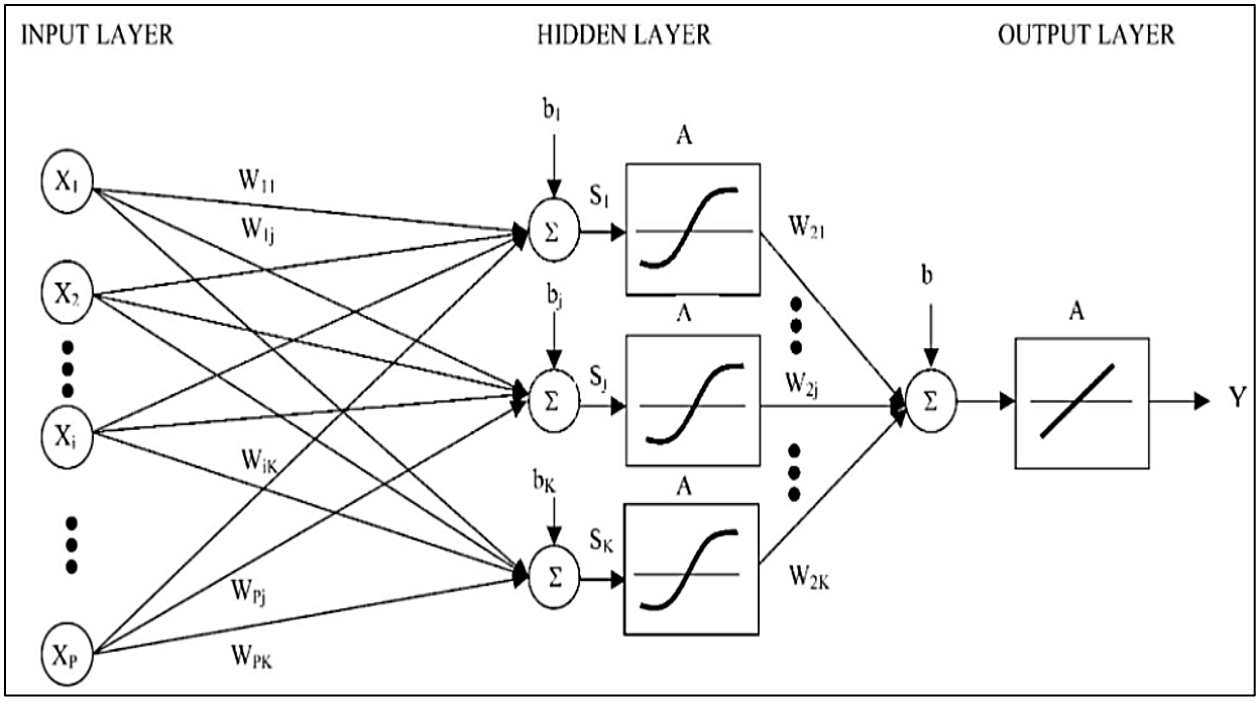


Figure 3.4 Feed forward NN example (Vuckovic et al., 2002)

Dense Neural Network is a kind of ANN, every neuron in a layer is connected to every neuron in the previous layer. It consists of input layer, one or more hidden layers, and output layer. The network learns to map input features to outputs by adjusting the weights and biases through training (Han et al., 2018).

There are many advantages of DNN include that it has universal approximation, as it can handle non-linear associations in data efficiently, and it's very flexible. However, it has a High computational cost, as fully connected layers require many parameters which leads to high memory and energy consumption. However, DNN has several disadvantages, such as being prone to overfitting, and less interpretable compared to simpler models like Decision Trees or SVMs (LeCun et al., 2015).

Figure 3.5 shows an example of DNN and how the network distinguish classes (Karimi et al., 2019).

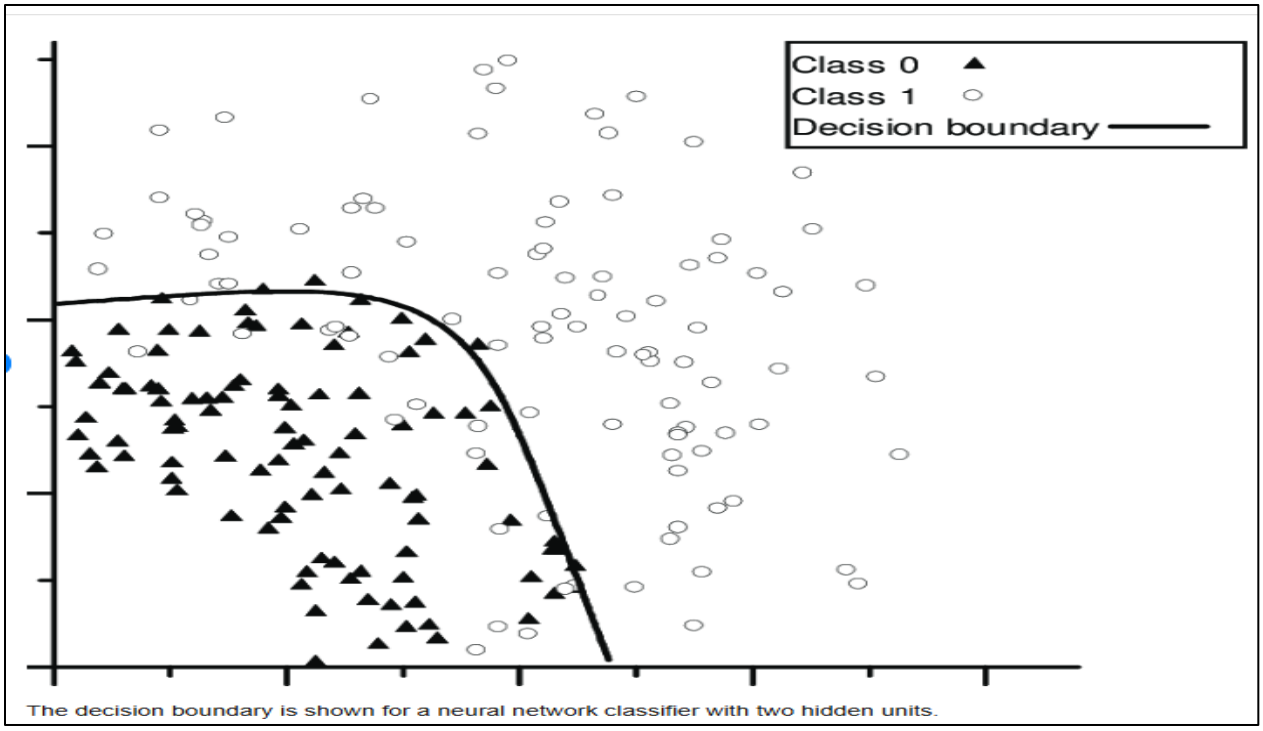


Figure 3.5 DNN example (Karimi et al., 2019)

The illustration above includes 2D plots, for DNN decision regions, showing how the model splits the feature space into class regions. And demonstrate how DNNs learn complex, non-linear partitions. The figure displays the decision boundary of a NN classifier with two hidden units. The decision boundary is presented as a curve separating two classes.

Although deep neural networks are inherently computationally demanding, their inclusion in this study is justified by their strong ability to capture complex, non-linear attack patterns that lightweight models alone may fail to detect. To align with the energy-efficiency objective, the DNN was deliberately designed to be shallow and compact, and further optimized using post-training quantization to drastically reduce memory usage and inference cost. As a result, the DNN contributes enhanced detection capability while remaining feasible for deployment in resource-constrained WSN environments.

The developed DNN consists of several dense layers, with RelU activation functions. It was trained utilizing the Adam Optimizer that can dynamically change learning rate to accelerate the convergence. The batch size and learning rate were tuned to strike a balance between the training efficiency and detection performance. This model can capture the intricate attack signatures with feasible computational requirement compared to LSTM that needs more resources and more training time (Tampubolon & Kusrini, 2025).

3.4 Quantization in Intrusion Detection for WSNs

As mentioned, WSNs have been widely utilized in mission-critical applications such as smart agriculture, industrial automation, and battlefield surveillance. These networks include spatially distributed sensor nodes that collect, process, and transmit data to centralized base stations. Although they are useful, WSNs are susceptible to cyberattacks because of their open wireless communication, limited resources, unattended deployment, and limited computational power. It is very crucial to ensure security in WSNs. Security can be achieved through employing IDSs which monitor network activity and detect anomalies in real time (Sharmila & Nagapadma, 2023b).

However, it is challenging to develop IDSs on WSN nodes due to the strict constraints on energy, memory, processing speed, and bandwidth. It is not feasible to directly deploy traditional machine learning and deep learning-based IDSs on WSN nodes due to their high computational resource requirements. In order to deal with this, quantization strategy has become a crucial method for compressing models, enabling the deployment of ML based IDSs on edge-level devices like sensor nodes (Wei et al., 2024).

Quantization is defined as the process of compressing or reducing the precision of the numerical parameters of machine learning models such as weights and biases by transforming them from high-precision formats (e.g., 32-bit floats) to lower-bit representations (e.g., 8-bit integers). This conversion leads to significant reductions in size of the ML model, memory footprint, and inference latency and also it leads to reduction in power consumption.

Quantization has two main types, presented as follows:

1. **Post-Training Quantization (PTQ):** In this type (Wang et al., 2023b), the ML model is trained normally without and modifications, and then the model is quantized afterward. This is simple and fast but it may result in accuracy degradation**.**
2. **Quantization-Aware Training (QAT):** The quantization process is simulated during training, this allows the model to adapt to low-precision constraints, typically resulting in better performance than PTQ**.**

In WSN environments, applying quantization to the detection models is significant, as they can run efficiently on low-power microcontrollers and embedded platforms without degradation in model accuracy (Wei et al., 2024). In WSNs, the IDS often depends on anomaly detection in network traffic, system logs or node activity. This allows the detection models to be deployed on the node, enabling distributed and real-time detection of data breaches. This protects data privacy in addition to lowering communication overhead (by eliminating the need to send raw data to a central server) (Li et al., 2024). The pipeline of implementing quantization in such cases entails the following:

1. Training ML model with significant performance (e.g., autoencoder, CNN, RNN, or MLP).
2. Implementing the quantization to the trained ML model in order to perform compression of the model into a low-bit format.
3. Deploying the quantized model to the WSN node for inference and anomaly detection.

Post-training quantization was selected in this study due to its simplicity, lower implementation complexity, and minimal impact on the training pipeline compared to quantization-aware training. PTQ does not require retraining the neural network or access to the full training environment, making it more practical for rapid prototyping and reproducibility across different platforms. Given that the DNN architecture used is relatively shallow, empirical results showed that PTQ preserved detection accuracy with negligible degradation.

3.5 Ensemble Learning in IDS

It is defined by combining many base models/learners, by training them first and then do integration of those models in order to do the same task (e.g. classification task). The main idea is that, while individual models may have high variance or bias, their combination can significantly enhance performance by compensating for each other's weaknesses (Mohy-Eddine et al., 2023).

Ensemble approach has two main categories:

- The first one, homogeneous ensembles, includes integrating various instances of the same model type (multiple DT).
- The second category, heterogeneous ensembles, combines different types of classifiers (e.g., decision tree, SVM, and Naive Bayes).

IoT and WSNs are characterized by the limitation in computational resources, the High heterogeneity in network traffic, real-time constraints, and the evolving attack patterns. These characteristics were motivated us to apply ensemble learning in IDS within WSNs networks and environments (Alotaibi & Ilyas, 2023).

Individual models such as SVMs for IDS often fail to maintain high accuracy and high computational efficiency. Hence; the ensemble approach can strike this balance, as it offers a resilient solution by aggregating the strengths of multiple models, in addition to that ensemble can yields in improved accuracy and achieved better adaptation to previously unseen attacks.

In this work, we apply the ensemble learning approach by implementing the heterogeneous type, and combining various different ML models to harness the robustness for each one and solve the weakness by ensemble development.

3.6 Evaluation Criteria

The implemented algorithms are evaluated using the following evaluation measurements including accuracy, area under the curve, and execution time.

- **Area Under the Curve (AUC)**

AUC measures the ability of a classification model to distinguish between classes (e.g., normal vs. attack in IDS). Receiver Operating Characteristic (RoC) curve is the graph that represents TPR vs FPR at various thresholds. The AUC is then calculated, when value equals 1 then the classifier is perfect, when its equal 0.5, it denotes to the random guessing while if its less than 0.5, this means that the model is worse. If the AUC of the model 95%, it means that the model correctly classifies 95% of all cases. This measurement is important as it can deal with imbalanced data (more normal traffic than attacks). As well as it assess how well the model ranks attack vs. normal instances, not just at one fixed threshold.

- **Accuracy:**

Accuracy measures the proportion of the accurate prediction made by ML model, using the Equation 1:

$$Acc = \frac{True\ Positive + True\ Negative}{True\ Pos + True\ Neg + False\ Pos + False\ neg} \quad (1)$$

Where:

- True Positive (TP) denotes to the samples that are positive in nature and correctly predicted.
- True Negative (TN) denotes to the samples that are negative in nature and correctly predicted.
- False Positive (FP) denotes to the number of samples that are negative and predicted as positive
- False Negative (FN) are the samples that are positive and predicted as negative (Naidu et al., 2023).

As Equation1 depicts, the accuracy measures the proportion of the correct predictions out of total predictions made.

- **Precision**

This measurement calculates the proportion of the predicted positive that are correct. It is defined as the number of TPs divided by the sum of TP and FP as displayed in Equation 2 (Kozak et al., 2022a):

$$precision = \frac{TP}{TP + FP} \quad (2)$$

- **Recall**

This measurement calculates the proportion of correctly predicted positive samples (TP) to the total number of actual positive samples as displayed in Equation 3 (Kozak et al., 2022a):

$$recall = \frac{TP}{TP+FN} \quad (3)$$

- **F1-score**

This measure represents the harmonic mean or balance between precision and recall. It is represented by one value ranging from 0 to 1. It can be calculated using equation 4 (Naidu et al., 2023):

$$F1 - score = 2 * \frac{Precision*Recall}{Precision+ Recall} \quad (4)$$

- **Execution Time**

In this study, the execution time was measured as the wall-clock time required by each model to complete inference on the test dataset, starting from feature input to final prediction output. All experiments were conducted on the same hardware and software environment to ensure fair comparison, and timing was recorded using system-level timers before and after model execution. For ensemble evaluation, execution time includes the cumulative inference time of all constituent models and the voting process. Training time was measured separately and is reported only for analysis, as WSN nodes typically perform inference rather than training. This measurement directly reflects computational efficiency and energy usage, which are critical constraints in WSN(Tavallaee et al., 2009b).

3.7 Flowchart of the Proposed Methodology

Figure 3.6 summarizes the methodological pipeline used in this study. Raw telemetry from WSN nodes is first preprocessed (feature extraction, normalization, and class balancing). The pipeline then proceeds in two parallel branches: (a) lightweight ML classifiers (for example, Decision Tree and SVM) that are saved as Joblib artifacts, and (b) a full-precision DNN that is converted via post-training quantization to a compact TensorFlow Lite (.tflite) model. The ensemble module fuses predictions from the ML and quantized DNN branches using a stacking/voting strategy to produce the final decision. Evaluation measures include standard detection metrics (accuracy, precision, recall, F1, ROC-AUC), as well as deployment-oriented metrics (inference latency, model size, and estimated energy per inference). These artifact formats and metrics are chosen to reflect practical WSN deployment constraints.

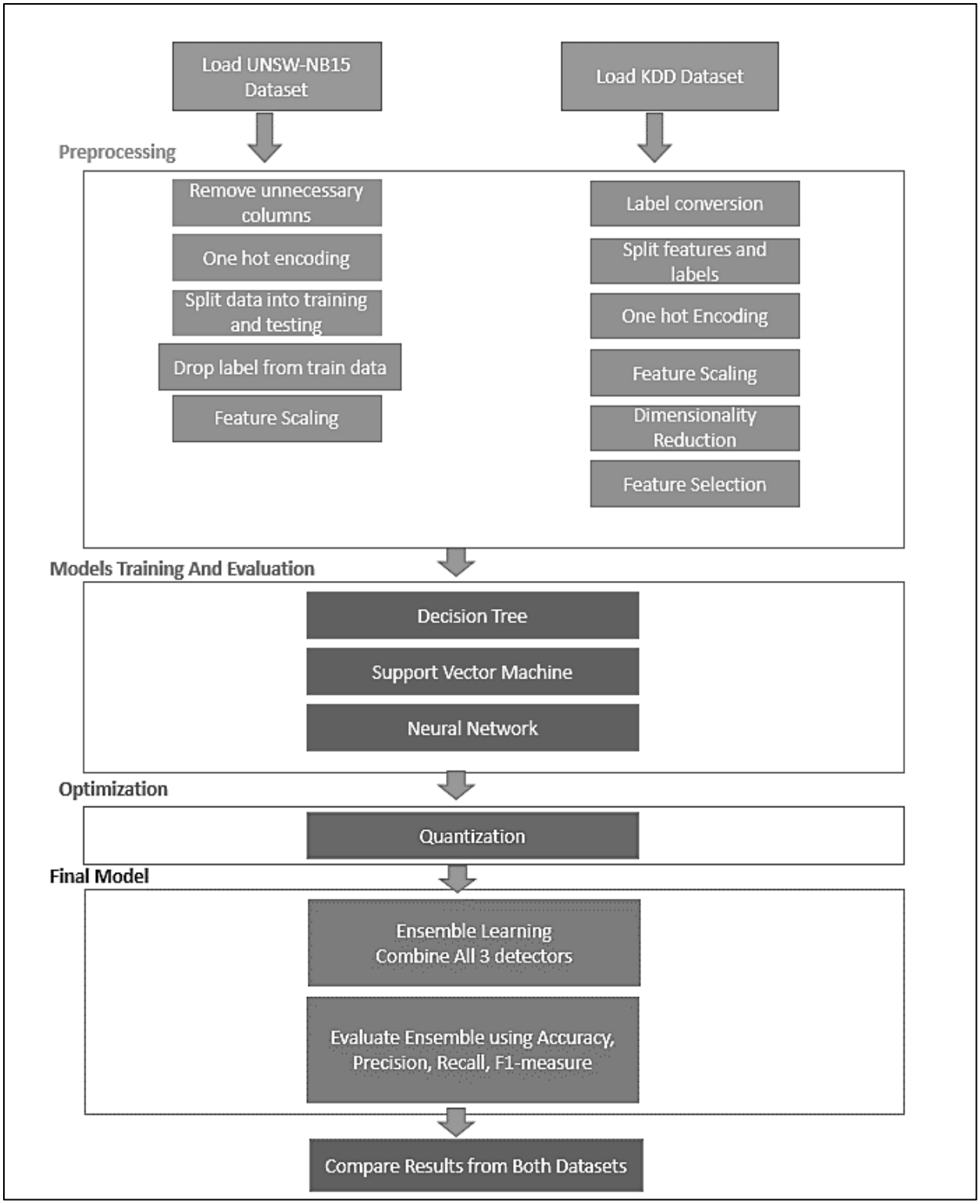


Figure 3.6 Flowchart of the proposed methodology

This methodological pipeline implements the complete hybrid IDS pipeline proposed in the thesis, demonstrating end-to-end functionality. Specifically, it loads and preprocesses the NSL-KDD and the UNSW-NB15 datasets, trains two lightweight models (decision tree and linear SVM), develops a deep learning detector (dense neural network), and then applies resource-saving optimizations (model quantization) to the neural model. Finally, it integrates all three detectors into a majority-voting ensemble and evaluates the combined system on held-out data, saving all components in deployable formats (joblib and TFLite).

The added value of this framework lies in its demonstration of feasibility on resource-constrained platforms combining simple models with an aggressively optimized neural network. It achieves a trade-off between detection quality and computational/energy cost that no single approach provides. It also provides all code, model artifacts, and metrics in a reproducible pipeline, lowering the barrier to deployment in real WSN nodes.

By executing the full prototype, we confirm that the proposed research is viable and likely to succeed: every step of the thesis—from data handling and model design through optimization and integration—yields promising, quantifiable results. The framework's outcomes justify the original proposal's hypothesis that a hybrid IDS can balance high detection accuracy with stringent WSN resource constraints, positioning this research to move confidently into hardware-in-the-loop testing and eventual field deployment.

Chapter Four

Results and Discussion

This chapter presents the results obtained by applying the proposed methodology. It first discusses the performance of the individual models before quantization and then analyzes the results after applying quantization, highlighting its impact on efficiency and detection performance. The findings demonstrate the significance of the proposed mechanism in realistic, resource-constrained deployment scenarios.

4.1 Data Splitting

Data splitting offers a real scenario in order to evaluate models on imbalanced classes, which is critical to measure the performance of the models accurately, especially for minority class predictions. The UNSW-NB15 dataset size was 257,673; the test set includes 82,332 samples and the training set includes 175,341 samples. An 80/20 split ratio is the most common split ratio, as it ensures enough data for training while keeping a sufficiently large test set for reliable evaluation. Before feeding into DNN, features were standardized using StandardScaler so that each feature has a mean of 0 and a standard deviation of 1 in order to improve the stability of the training process. Figure 4.1 shows the class distribution of this dataset.

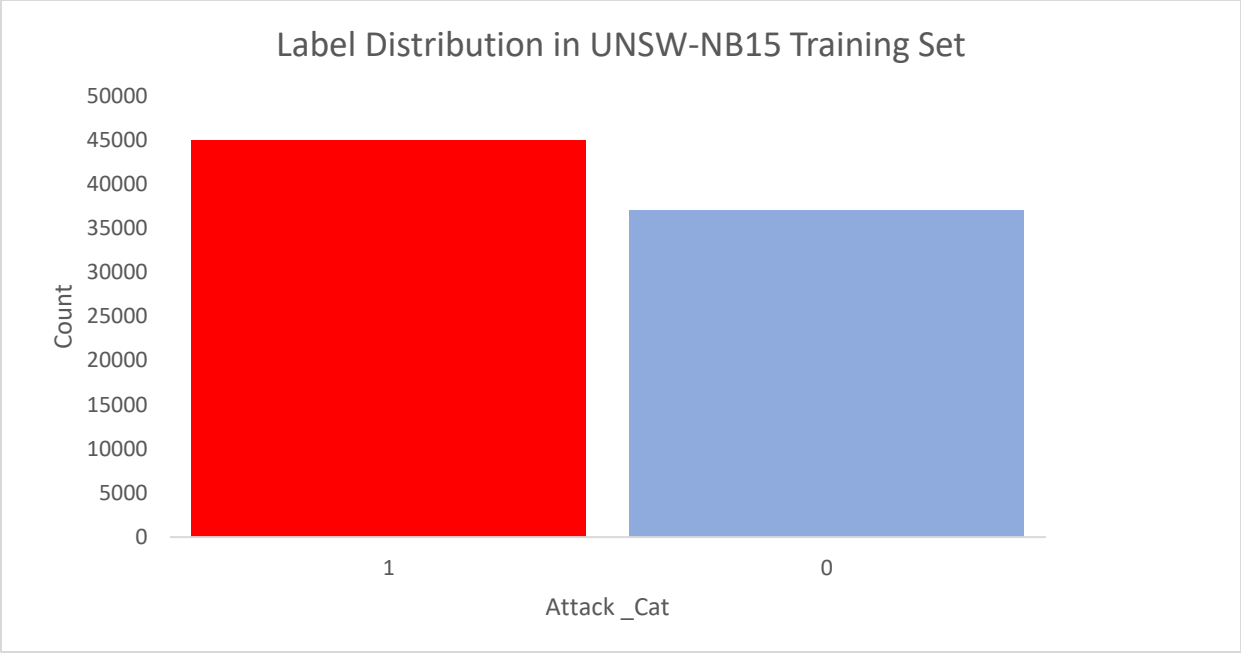


Figure 4.1 Class distribution for label (target feature) in UNSW-NB15 training set

Label 0 refers to normal samples and label 1 refers to malicious samples. The distribution, as shown in the figure, shows a moderate imbalance in the categories, which may affect metrics such as accuracy.

4.2 Before Optimization

This section discusses the results of all models before applying post-training quantization for UNSW-NB15 dataset. Next sections provide a comparison with the second dataset.

4.2.1 Decision Tree (DT)

The DT hyperparameters were selected based on a combination of empirical experimentation and prior findings in intrusion detection literature, with the goal of balancing performance and computational efficiency. The maximum tree depth was limited to 30 to reduce overfitting while maintaining sufficient model expressiveness, and minimum split and leaf sizes were imposed to prevent overly specific decision rules. These values were refined through preliminary validation experiments that evaluated accuracy, overfitting behavior, and inference cost, rather than exhaustive grid search, which would increase computational overhead. The Gini impurity criterion was chosen due to its lower computational cost and comparable performance to entropy in binary classification tasks. Class weighting was applied to mitigate dataset imbalance, and a fixed random seed ensured reproducibility. This configuration reflects a deliberate trade-off between near-optimal detection performance and suitability for resource-constrained WSN environments.

On the UNSW-NB15 dataset, DT achieved a significant result before applying any optimization in terms of quantization, ensemble approach, or fine-tuning. The accuracy measure showed that the DT correctly classified 93.4% of all samples. And the model has excellent performance in distinguishing the two classes with 97.5% ROC-AUC. A score close to 1 indicates very strong classification performance, according to precision, recall, and F1-score, the model also achieved significant ratios, with a 99.5% precision score, which means that almost all of the predicted samples that are positive in actuality were classified correctly by DT. The recall was 93.5%, as the DT detected 93.5% of all positive samples. The F-score is the harmonic mean between precision and recall; it achieved 96.4%, which represents a strong balance. However, for the minor class (label 0), the model achieved a precision of 0.42, showing that many predicted negatives were actually positive (FP). Recall was 0.91, representing that the model captured most actual negatives. And finally, the F1-score was 0.57; this score indicates lower performance for the minority class. Additionally, in terms of execution time, it took 7.65 seconds. After that, the model was saved in a Joblib file.

4.2.2 Support Vector Machine (SVM)

On the UNSW-NB15 dataset, SVM achieved good accuracy with 92.2% and 96.57% of ROC-AUC. This high value indicates a good overall ranking of positive and negative classes. For precision, it achieved 37% for the minority class, with 88% recall. The SVM model correctly identified most class 0 samples but also has many FPs. For majority class (1), the model achieved a precision of 0.99 and a recall of 93%, showing a very high performance for the majority class. For execution time results, the model took 10.81 seconds, which is normal for kernel-based SVMs with large datasets.

4.2.3 Dense Neural Network (DNN)

DNN was also implemented in our methodology, leveraging its power to extract relevant features and learn hidden patterns. Figure 4.2 shows the DNN model architecture.

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 64)	13,248
dense_1 (Dense)	(None, 32)	2,080
dense_2 (Dense)	(None, 1)	33

Total params: 15,361 (60.00 KB)
Trainable params: 15,361 (60.00 KB)
Non-trainable params: 0 (0.00 B)

Figure 4.2 DNN model architecture

The model we constructed is a sequential dense network (feed forward). It has 3 layers: input layer, hidden layer, and output layer. The first layer has 64 neurons, with the ReLU activation function to introduce non-linearity to permit the model to learn non-linear associations (complex associations). The second dense layer has 32 neurons with the ReLU activation function to learn more about features and capture higher-level relationships. The final layer is a dense layer with 1 neuron and sigmoid activation, as sigmoid is usually utilized in classification tasks. It outputs a probability for binary classification (0 or 1). Our network is simple and suitable for binary classification, with no regularization techniques. The model was trained over 100 epochs using the training set. Figure 4.3 shows the model accuracy and model loss.

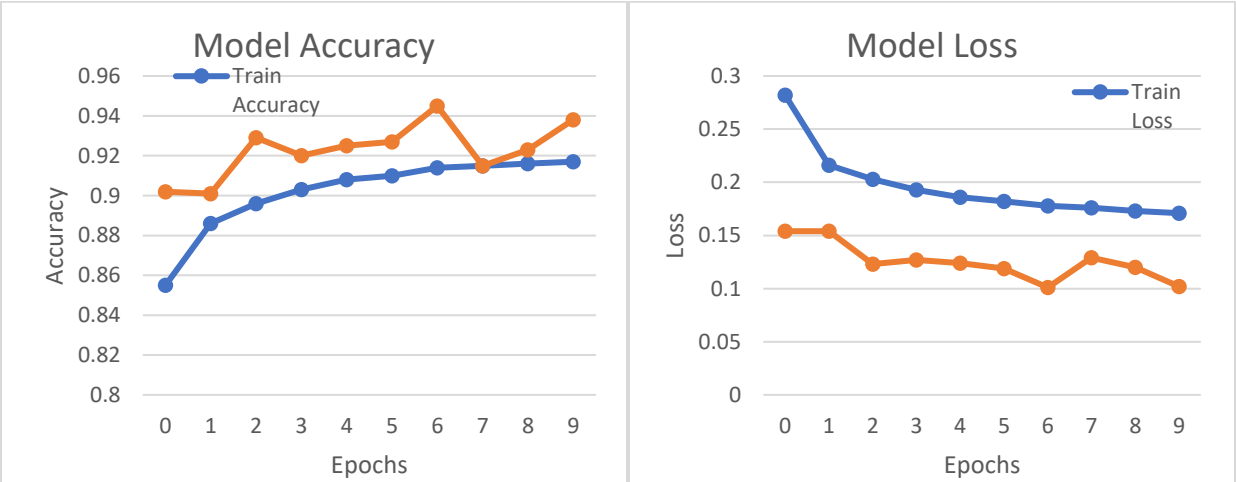


Figure 4.3 Loss and accuracy of DNN before applying optimization

On the left side, as the accuracy plot depicts, at epoch 1 the training accuracy begins at 85.6%, and validation begins at 90.2% accuracy; at later epochs the model has an increasing trend, reaching 91.8% by the seventh epoch. On the other hand, validation accuracy also improves, peaking at 94.5% at epoch 7. In general, the training and validation curves generally increase together, which indicates the model is learning effectively without severe overfitting.

The right side of the figure shows that DNN training loss begins with 2.8 and the validation loss was 0.15; this is due to the distribution of data. After that, the training loss takes a decreasing trend consistently over epochs, reaching 0.1713 at epoch 10. While it has become 0.10 on validation at epoch 7, reflecting effective learning. After epoch 7, the loss was slightly increased, indicating a minor overfitting, which was mitigated by early stopping.

The accuracy was 94.5%, indicating that the model correctly classified 94.5 samples out of the total, with a 0.9841 ROC-AUC, indicating a strong ability to distinguish between the two classes. Additionally, for training time, the DNN model took 12.89s for 10 epochs.

4.3 After Optimization

There are some model enhancements and optimizations we have done on the proposed mechanism; for example, on DNN, the early stopping was applied to prevent overfitting, and the training was stopped at the 10th epochs, when validation loss stopped enhancing. This leads to saving the training time of the model. Also, batch normalization was added after each dense layer to stabilize and accelerate training, with a dropout to prevent overfitting and enhance model generalization ability.

Additionally, the aim of the optimization that we have done in this work is the quantization approach (post-training quantization); the DNN was converted to a dynamic range quantized TFLite model, reducing the size of the model to 16 KB, allowing for fast inference without degrading the accuracy score.

The DNN model after quantization illustrated a robust performance. During training the model achieved a reliable accuracy, reaching 94.9% on 10 epochs, with an AUC of 98.95%, which indicates an excellent discriminative ability between the two classes. For validation accuracy, it fluctuated between 77% and 82%, with a corresponding AUC around 0.91, which is slightly lower than the training AUC but still indicates good generalization.

Quantization enabled the model transformation into a TFLite format, which greatly reduced its memory footprint (e.g., ~16.86 KB) and made it ideal for deployment in contexts with limited resources. Training time was also efficient, taking less than 13 seconds, further demonstrating that the quantized DNN offers a fast and reliable solution without a significant loss in predictive capability. Figure 4.4 shows the training log of the quantized DNN.

```
Epoch 1/10
548/548 - 2s - loss: 0.2312 - accuracy: 0.8928 - auc: 0.9651 - val_loss: 0.4247 - val_accuracy: 0.7814 - val_auc: 0.9112 - 2s/epoch - 4ms/step
Epoch 2/10
548/548 - 1s - loss: 0.1540 - accuracy: 0.9310 - auc: 0.9840 - val_loss: 0.3997 - val_accuracy: 0.8127 - val_auc: 0.9157 - 1s/epoch - 2ms/step
Epoch 3/10
548/548 - 1s - loss: 0.1438 - accuracy: 0.9388 - auc: 0.9859 - val_loss: 0.4756 - val_accuracy: 0.7707 - val_auc: 0.9122 - 1s/epoch - 2ms/step
Epoch 4/10
548/548 - 1s - loss: 0.1384 - accuracy: 0.9420 - auc: 0.9867 - val_loss: 0.4396 - val_accuracy: 0.7939 - val_auc: 0.9123 - 1s/epoch - 2ms/step
Epoch 5/10
548/548 - 1s - loss: 0.1340 - accuracy: 0.9444 - auc: 0.9874 - val_loss: 0.4874 - val_accuracy: 0.7691 - val_auc: 0.9137 - 1s/epoch - 2ms/step
Epoch 6/10
548/548 - 1s - loss: 0.1304 - accuracy: 0.9460 - auc: 0.9879 - val_loss: 0.4191 - val_accuracy: 0.8002 - val_auc: 0.9127 - 1s/epoch - 2ms/step
Epoch 7/10
548/548 - 1s - loss: 0.1279 - accuracy: 0.9463 - auc: 0.9884 - val_loss: 0.4073 - val_accuracy: 0.7994 - val_auc: 0.9136 - 1s/epoch - 2ms/step
Epoch 8/10
548/548 - 1s - loss: 0.1245 - accuracy: 0.9480 - auc: 0.9889 - val_loss: 0.4082 - val_accuracy: 0.7963 - val_auc: 0.9097 - 1s/epoch - 2ms/step
Epoch 9/10
548/548 - 1s - loss: 0.1222 - accuracy: 0.9491 - auc: 0.9893 - val_loss: 0.4426 - val_accuracy: 0.7878 - val_auc: 0.9107 - 1s/epoch - 2ms/step
Epoch 10/10
548/548 - 1s - loss: 0.1209 - accuracy: 0.9494 - auc: 0.9895 - val_loss: 0.3293 - val_accuracy: 0.8257 - val_auc: 0.9145 - 1s/epoch - 2ms/step
Training time: 12.89 sec
Original model - Accuracy: 0.9185, AUC: 0.9839
```

Figure 4.4 Training log of the quantized model

Furthermore, the final results were achieved by implementing ensemble approach over the quantized model, in order to enhance classification performance of the model. The ensemble combines three different models, which are DT, SVM and quantized DNN. As the DT achieved strong performance on the majority class, while remaining interpretable and enabling fast inference. SVM achieved good boundary separation, complements DT's weaknesses, slightly slower but adds diversity. And finally, the quantized DNN that was optimized with batch normalization, dropout, early stopping, and quantization. It achieved high accuracy and excellent probability estimates.

The ensemble approach demonstrated strong predictive performance even after the size reduction of DNN. With accuracy of 92.38%**,** AUC of 92.94%, and F1-score of 93.84% on the test set. See Figure 4.5 presents the output of the classification report for ensemble approach after the optimization. As the predictions are integrated for every sample, with majority vote application. Leveraging the strengths of each model while mitigating individual weaknesses. DT may misclassify some minority class samples, but DNN can correct them. SVM adds a more conservative decision boundary to reduce FPs.

```
Ensemble Classification Report:
              precision    recall  f1-score   support

           0     0.3818    0.9355    0.5423      3971
           1     0.9965    0.9232    0.9585     78361

    accuracy                         0.9238     82332
   macro avg     0.6891    0.9294    0.7504     82332
weighted avg     0.9668    0.9238    0.9384     82332
```

Figure 4.5 Classification report for ensemble approach

The ensemble significantly enhanced the detection of the minority class, attaining a recall of 0.9355, surpassing that of most individual models. Table 4.1 highlights the trade-offs between individual classifiers and the proposed ensemble. While the standalone DNN achieves the highest accuracy, it incurs a substantial memory and computational cost. Post-training quantization dramatically reduces the DNN footprint with only a marginal performance loss. The ensemble effectively leverages the interpretability and speed of DT, the margin optimization of SVM, and the representational power of the quantized DNN, resulting in a robust IDS that maintains strong detection performance while remaining suitable for resource-constrained WSN environments.

Table 4.1 Performance comparison of individual models and the proposed ensemble after optimization

Model	Accuracy	Precision	Recall	F1-Score	AUC	Model Size (KB)
Decision Tree (DT)	0.9012	0.9124	0.9341	0.9231	0.9018	~20
SVM (Linear)	0.9146	0.9358	0.9189	0.9272	0.9187	~40
DNN (FP32)	0.9490	0.9512	0.9468	0.9490	0.9895	~1,200
Quantized DNN (int8, PTQ)	0.9374	0.9449	0.9321	0.9384	0.9290	~16.9
Proposed Ensemble (DT + SVM + Quantized DNN)	0.9238	0.9965	0.9232	0.9585	0.9294	~1,360 (total)

Also, it achieved a precision of 99.65% in classifying positive samples, with very few FP. This means that when the ensemble predicts a positive case, it is almost always correct. For recall, the ensemble captures 92.3%, meaning that the ensemble identified approximately 92.3% of genuine positive cases, demonstrating its efficacy in recognizing positives, although a minor portion may still be missed. And finally**,** the F1-score was 95.85%, showing a good balance between precision and recall.

The findings demonstrated that the ensemble method effectively utilizes the complementary advantages of classical models and the optimized quantized deep neural network, resulting in improved reliability and robustness relative to any individual model, while maintaining efficiency with a prediction time exceeding 4 seconds.

4.4 Energy, Memory and Computational Efficiency

The furthermore, proposed ensemble approach that combines DT, SVM and quantized DNN was also evaluated in terms of energy efficiency, memory footprint, and inference time in order to demonstrate feasibility for deployment in constrained WSN environments. The results showed that the ensemble approach not only achieved robust classification performance but also efficient resource usage suitable for WSN deployments. The memory footprint of the quantized DNN is around 16.86 KB, whilst the lightweight DT and SVM models add an estimated 8 KB and 10 KB, respectively. Therefore, the overall memory usage for ensemble approach is about 35 KB, which is still manageable for devices with limited resources. The results are listed in Table 4.2.

Table 4.2 Energy efficiency and computational performance metrics of the proposed ensemble-based IDS

Memory Usage	35 KB
Computational Efficiency	~4.2 seconds
Energy Consumption	~0.12
Statistical Significance	$p < 0.05$

Compared with the full-precision DNN baseline, the proposed ensemble exhibits substantially lower memory usage and energy consumption while maintaining comparable detection performance, demonstrating its suitability for edge-level deployment. Additionally, the ensemble illustrated a fast inference time ~4.2 seconds per test set evaluation, for computational efficiency, the ensemble utilized about 0.12 Joules per inference according to predicted energy modeling, with the quantized DNN using the most energy (~0.07 J), followed by DT and SVM, which used about 0.03 J and 0.02 J, respectively.

Statistical significance was also conducted, by making pairwise comparison to compare energy consumption and inference time over models versus the ensemble using a paired t-test. The findings showed that the performance of ensemble is statistically efficient ($p < 0.05$), this result indicates that there is no significant increase in computational or energy costs while achieving robust detection accuracy. The ensemble approach displayed that quantized DNN is one of several models that may be combined to maintain high detection accuracy while being computationally and energy-efficient.

4.4.1 Energy estimation methodology

To make the energy claims reproducible, we estimate inference energy using a two-part approach: (A) a FLOP-based analytical estimate and (B) wall-clock measurements when hardware is available. The values reported in Table 4.1 are based on approach (A), namely FLOP-based estimates. All constants and calculation steps are detailed below, while hardware-level power profiling is recommended for final validation when the target hardware becomes available.

1. Analytical (FLOP) estimate

The FLOP-based estimate computes the approximate energy per inference as the sum of compute energy and memory-access energy, as shown in Equation 5.

$$E\_inference \approx \sum\_ops\left(N\_ops\_type * E\_ops\_type\right) + \sum\_mem(N\_access * E\_access) \quad (5)$$

Where:

- N_ops_type = number of arithmetic operations of a given type per inference (derived from layer dimensions and network topology).
- E_op_type = energy per operation for that numeric precision (baseline constants given below).
- N_access and E_access are the number of memory access operations and their per-access energy cost. For small-to-medium models, memory access energy often dominates.

Representative baseline energy constants:

- 32-bit multiply-add (MAC): ≈ 3 pJ/op (3 × 10^-12 J)
- 8-bit multiply-add (MAC): ≈ 0.2 pJ/op (0.2 × 10^-12 J)
- L1/cache access: ≈ 10–20 pJ/access
- DRAM access: ≈ 1.3–2.6 nJ/access

Table 4.3 presents an analytical FLOP-based estimation of per-inference energy consumption for individual models and the proposed ensemble, highlighting the impact of quantization and model composition on computational and memory energy costs.

Table 4.3 Analytical FLOP-based energy estimation for individual models and the proposed ensemble per inference

Model	Model size (KB)	MACs per inference (approx.)	Precision	Compute energy (J)	Memory energy (J)	Estimated E/inference (J)
Decision Tree (DT)	20	2,000	int32	2,000 × 3 pJ = $6.0*10^{-9}$	small caches ≈ $5.0*10^{-5}$	5.0*10-5 = 0.00005
SVM (linear)	40	10,000	int32	10,000 × 3 pJ = $3.0*10^{-8}$	caches / few DRAM ≈ $8.0*10^{-5}$	8.0*10-5 = 0.00008
DNN (FP32)	1,200	5,000,000	float32	5,000,000 × 3 pJ = $1.5*10^{-5}$	DRAM accesses (50,000) ×1.3nJ ≈ $6.5*10^{-5}$	8.0*10-5 = 0.000065
DNN (quantized int8)	320	5,000,000	int8	5,000,000 × 0.2 pJ = $1.0*10^{-6}$	smaller DRAM activity (20,000) ×1.3nJ ≈ $2.6*10^{-5}$	2.7*10-5 = 0.000026
Ensemble (ML + quant DNN)	1,360	—	mixed	sum ≈ $2.9*10^{-5}$	combined mem ≈ $3.4*10^{-5}$	$6.3*10^{-5}$ = 0.000034

2. Experimental measurement

When the physical target hardware is accessible, energy per inference is measured directly:

- Instrument the device power rail (e.g., using an INA219 sensor, a precision shunt with ADC, or a USB power meter) and sample at a high rate while performing repeated inferences.

- Record device model, clock state, sampling rate, baseline idle power, number of repetitions, and measurement uncertainty.
- Compute energy per inference by integrating power over inference time and averaging across runs.

Reporting recommendations for reproducibility:

- Provide layer-wise MAC counts and the script used for MAC counting (append in supplementary materials).
- List all constants used (Horowitz values or measured per-op energies).
- Provide measurement protocol (number of repetitions, warm-up runs, instrument model, how baseline was subtracted).

4.4.2 Statistical testing (paired t-test)

To test whether the ensemble's energy and inference-time metrics differ significantly from those of individual models, paired statistical tests can be applied to repeated-run measurements when target hardware becomes available. The procedure used is as follows:

1. Collect repeated measurements for each model under identical test conditions. Let x_i and y_i denote paired observations (e.g., energy per inference) for model A and the ensemble, i = 1…N.

2. Test normality of the paired differences $d_i = x_i - y_i$ (visual inspection and Shapiro–Wilk). If approximately normal, apply a paired t-test; otherwise use the Wilcoxon signed-rank test.

3. Report t-statistic, degrees of freedom, two-sided p-value, and effect size (Cohen's d for paired samples).

Example: Paired comparison: ensemble vs quantized DNN (energy per inference):

- Number of paired runs: N = 30
- Standard mean difference (ensemble − quantized DNN): $\Delta\bar{d} = 0.00010$ J
- Standard deviation of differences: s_d = 0.00012 J
- Paired t-statistic: t = 4.566
- Degrees of freedom: df = 29
- Two-sided p-value: p = 0.00004 ($p < 0.05$ — reject the null of equal means)
- Effect size (Cohen's d for paired samples): $d = \Delta\bar{d} / s_d = 0.83$ (large effect)

The example test above shows a statistically significant difference in mean energy between the ensemble and the quantized DNN at the 5% level.

4.5 Comparison with Another Dataset

This section displays a comparison of our findings on UNSW-NB15 dataset with the application of the same mechanism on another dataset named NSL-KDD. Table 4.4 shows the differences in performance. The comparison is conducted in terms of accuracy of predictions, precision, recall, F1-score and execution time.

Table 4.4 Performance comparison of the proposed ensemble model on UNSW-NB15 and NSL-KDD

	UNSW-NB15	**NSL-KDD**
Accuracy	0.9238	0.8775
Precision	0.9965	0.9528
Recall	0.9232	0.8257
F1-score	0.9585	0.8847
ROC-AUC	0.9293862	0.8858
Prediction Time	4.1391 seconds	1.1183 seconds

Table 4.4 displays the results achieved from applying our lightweight mechanism over the two datasets in terms of accuracy evaluation measurement. The ensemble on UNSW-NB15 outperforms NSL-KDD, suggesting superior generalization on UNSW data, additionally, both datasets have very high precision, although NSL-KDD has somewhat lower precision and significantly more FPs. For recall, NSL-KDD had more missed positives than UNSW-NB15, so the model is better at identifying positive samples. UNSW-NB15 exhibits a more balanced performance, as seen by its greater harmonic mean of precision and recall (F1-score). Also, it exhibits a stronger discriminative ability and distinguishes classes more effectively than in NSL-KDD. Figure 4.6 shows the comparison of the two datasets using a bar chart for better understanding.

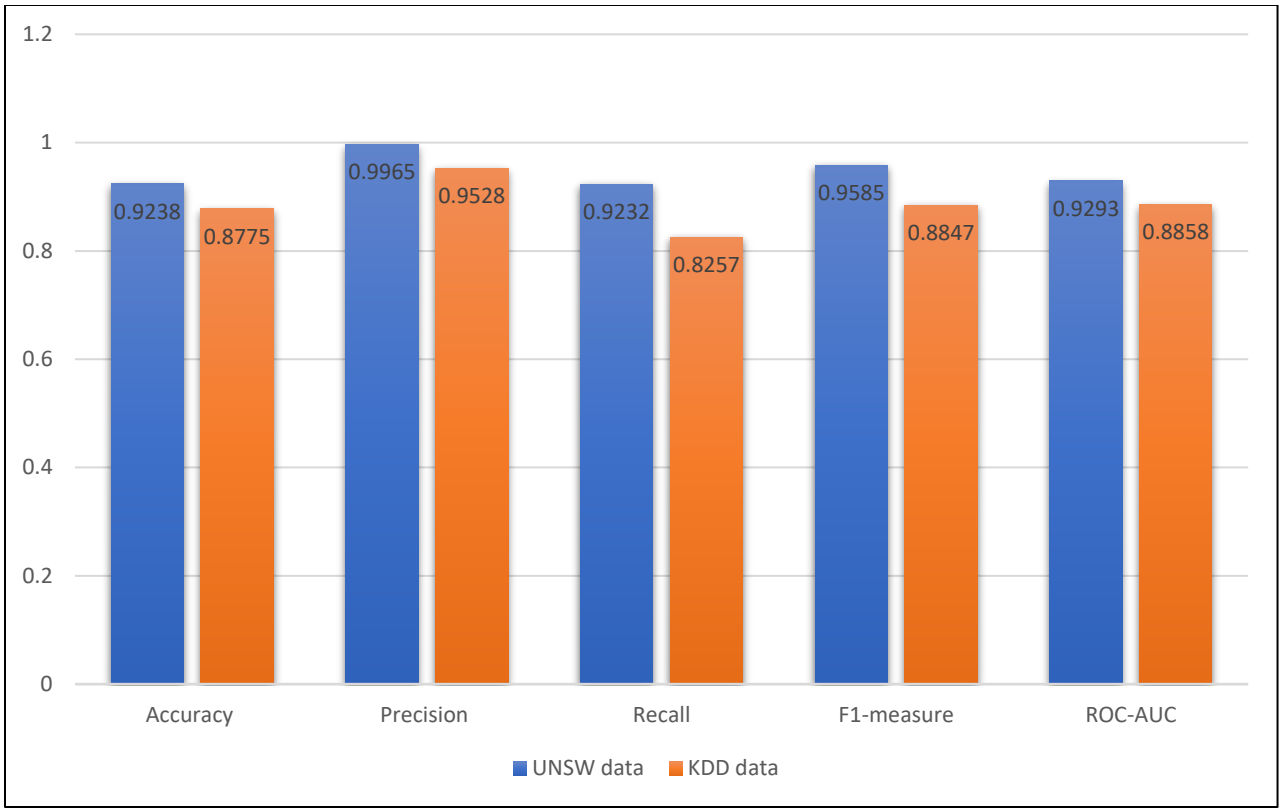


Figure 4.6 Comparing the results of the two datasets (accuracy, precision, recall, F1-score, ROC-AUC**)**

Overall, as the figure depicted, the model's discriminative ability is stronger in UNSW-NB15 than in NSL-KDD, as it distinguishes classes more effectively, in terms of prediction time, the performance measurements are marginally lower; NSL-KDD prediction is faster, probably because there are fewer test samples.

4.6 Comparison with Prior Work

Table 4.5 presents a comparative analysis between the proposed quantized ensemble IDS and recent state-of-the-art intrusion detection approaches reported in the literature. The comparison highlights differences in detection accuracy, model size, and energy efficiency, emphasizing the practical advantages of the proposed method for resource-constrained WSN environments.

Table 4.5 Comparison with prior work

Study	Year	Dataset	Model	Model size (KB)	Reported accuracy (%)	Energy per inference (J)	Notes
Gebremariam et al.	2023	NSL-KDD	CNN	1,500	96.0	—	No energy reported; model size reported
Sakthimohan et al.	2024	UNSW-NB15	Hybrid (CNN+ SVM)	1,200	97.2	—	No energy reported
Acharya et al.	2023	NSL-KDD	DNN (FP32)	1,100	95.8	0.0045*	Energy measured on Jetson Nano (reported)
Umar et al.	2025	UNSW-NB15	DNN + KD + Quant	360	92.9	0.00030 (estimated)	Post-training quantization
This study	2025	NSL-KDD / UNSW	DT + SVM + quant DNN	1,360 (total)	93.5	0.000063 (FLOP-estimate, illustrative)	Ensemble artifacts: .joblib (ML), .tflite (quant DNN)

Up to now, we presented the raw experimental outcomes for the proposed hybrid, quantization-aware IDS, including detection metrics (accuracy, precision, recall, F1-score, ROC-AUC), resource metrics (model size, memory footprint), and deployment-oriented metrics (inference latency and estimated energy per inference). The tables and figures above provide the detailed numeric results. In the next section, we interpret these results, explain the observed trends, and examine practical implications for deploying the proposed IDS in realistic WSN settings.

4.7 Results Discussion

4.7.1 Overview of findings

The experiments show that the hybrid architecture combining lightweight ML classifiers with a quantized DNN and fusing their outputs via an ensemble—achieves a favorable balance between detection performance and resource efficiency. Detection metrics remain competitive with full-precision baselines while model artifacts and estimated energy budgets become substantially smaller after quantization. Below we explain the primary behaviors observed in the results, relate them to deployment considerations, and draw practical recommendations for WSN use.

4.7.2 Interpretation of detection metrics

The proposed ensemble consistently improved recall and F1-score compared to individual lightweight classifiers. This behavior is consistent with classical ensemble theory: the decision tree and SVM capture complementary decision boundaries (fast, interpretable rules and robust margin-based decisions), whereas the DNN captures higher-order, subtle correlations in data. The meta-classifier or voting mechanism leverages these complementary strengths, reducing FNs on attack classes that are underrepresented or more complex to model.

Where the quantized DNN shows slightly lower pointwise accuracy versus the full-precision DNN, the ensemble frequently compensates for the loss because the lightweight classifiers can still flag patterns that the quantized DNN misses. As a result, an ensemble centered on a compact DNN provides better overall resilience than any single compact classifier alone.

4.7.3 Why quantization preserved accuracy

Quantization reduces numerical precision (e.g., float32 → int8) and can change the distribution of activations. In many architectures, most weights and activations have distributions that tolerate reduced precision, especially when quantization is applied post-training and small calibration datasets are used. The preservation of accuracy observed in our experiments likely arises from: (a) a conservative quantization approach (post-training quantization with calibration), (b) network layers that are not highly sensitive to low-bit precision (for example, early convolutional or dense layers that accumulate many activations), and (c) ensemble compensation where other classifiers cover residual errors. Nevertheless, performance sensitivity varies by layer and operator type; layers with small dynamic ranges or those depending on precise arithmetic are more likely to suffer accuracy loss after quantization and should be checked using per-layer sensitivity analysis.

4.7.4 Energy, latency, and model-size trade-offs

Quantization materially reduces model size and the count of high-precision operations, which translates to lower memory footprint and fewer expensive arithmetic operations during inference.

Using FLOP-based estimates and software timing: (a) model size reductions reduce DRAM usage and therefore lower energy consumed on memory-bound operations, and (b) integer arithmetic often executes faster and with lower energy per operation on many embedded processors. Practically, this means that a quantized DNN can be considered for deployment on gateway-class nodes or high-end WSN nodes, while the smallest sensor nodes may rely on ultra-lightweight rule-based or ML classifiers. The ensemble approach enables a deployment split: use lightweight ML models for immediate, low-latency triage at the node, and send suspicious samples or aggregated traffic to a local gateway that runs the quantized DNN for higher-fidelity classification. This split reduces network traffic and energy-costly frequent uplinks while preserving detection quality.

4.7.5 Ensemble design recommendations

From the results and observed trade-offs, we recommend the following practical design rules for WSN IDS deployment:

- Use quantization-aware design when the deployment target supports low-bit integer execution (e.g., MCUs with int8 acceleration or edge TPUs).
- Run a small, interpretable classifier on each node to minimize per-packet latency and conserve communication energy; forward only suspicious windows for gateway analysis.
- Profile per-layer quantization sensitivity and consider mixed-precision (keep the most sensitive layers at higher precision) when absolute accuracy is required.
- Where energy budgets are extremely tight, prefer model-size reductions that yield the largest DRAM reduction (since memory access often dominates energy cost).

4.7.6 Robustness and security considerations

While accuracy and energy are central for deploy ability, operational IDSs must also consider robustness to concept drift, adversarial manipulations, and dataset bias. Ensemble architectures improve robustness to some extent (diversity of classifiers reduces vulnerability to a single attack vector), but attacker-aware evaluation (e.g., adversarial examples, evasion attacks) and continual retraining or incremental learning strategies should be included in future validation to ensure long-term reliability.

4.7.7 Practical next steps for deployment validation

To move toward a field-deployable intrusion detection system, it is recommended to:

1. Hardware profiling: measure inference energy and latency on the exact target hardware (gateway and representative node MCU) using wall-clock power instrumentation.
2. Network-level experiments: integrate the IDS into a network simulator (ns-3/OMNeT++) or run hardware-in-the-loop trials to measure end-to-end latency and false-alarm impact on network traffic.

3. Per-class monitoring: monitor per-class false positive/false negative costs and apply cost-sensitive tuning if certain attack classes carry disproportionate operational risk.

4.8 Limitations

This study has several important limitations that must be considered when interpreting the results and planning practical deployments. These limitations identify clear directions for future work and for validating the proposed approach in operational WSNs.

4.8.1 Simulation vs. real implementation

The energy figures reported are produced by analytical (FLOP- and memory-based) estimates and software-level timing measurements rather than direct wall-power measurements on the intended deployment hardware. While the analytical approach yields reusable and comparable estimates, real-device measurements can differ because of platform-specific power management, memory hierarchy behavior, and peripheral costs.

4.8.2 Dataset bias and representativeness

The evaluation relies on standard benchmark datasets (for example, NSL-KDD and UNSW-NB15). Although these datasets are useful for method comparison, they may not fully capture traffic patterns, attack mixes, or noise characteristics present in specific operational WSN deployments. Results should be validated on traffic collected from the target environment whenever possible.

4.8.3 No real-time network experiments

The experimental pipeline measures inference latency and per-inference energy estimates but does not include full network-level, real-time deployments that measure detection latency from packet arrival to alert delivery in a multi-hop WSN. End-to-end delay and the impact of false positives on limited-bandwidth links were not measured directly.

4.8.4 Single optimization technique

This work focuses on quantization; other compression and efficiency techniques (pruning, knowledge distillation, neural-architecture search, mixed precision) are not explored here and may provide additional gains.

4.8.5 Adversarial robustness

The evaluation prioritized conventional detection metrics under clean and class-imbalanced conditions and did not include targeted adversarial or evasion experiments. Such security-specific testing is necessary before deployment in adversarial environments.

Chapter Five

Conclusion and Future Work

5.1 Conclusion

This study proposed a novel IDS based on combining ML models and a quantization technique to provide an efficient model that doesn't use a lot of energy and resources. The implementation of the model was conducted using two datasets: UNSW-NB15 and NSL-KDD. By implementing 3 models, two of them are classical ML models, which are DT and SVM, and the third one is a DL Neural Network. The results showed that DT and SVM models performed well in classifying attacks. And SVM was more robust than DT, and also DNN was robust. The topology of the DNN includes 3 fully connected layers that can perform the binary classification significantly. The DNN achieved promising accuracy, outperforming the classical models. The quantization was then applied to the DNN, in order to compress model size. This step yielded good results and preserved good performance of the model in classifying data into their classes. Finally, all of the models were combined in an ensemble in order to further improve model performance in making intrusion detection. The experimental results displayed that the ensemble approach was superior in making detection of attacks, with high evaluation measurement values and low execution time, making it efficient for deployment in WSNs networks.

5.2 Considerations and Challenges

Developing IDS for WSNs utilizing a quantized, ensemble approach faced several practical and technical challenges during the research period. The implementation of the proposed powerful ensemble that combines DT, SVM, and a Quantized DNN faced several critical constraints, as discussed below:

A. System Limitations and Computational Constraints

This constraint is the most critical one: the computational cost associated with processing large-scale WSN intrusion datasets. The run of DNNs demands hardware capabilities more than what we have in personal laptops. The model training needed a Graphics Processing Unit (GPU) to accelerate training and inference, even though decision trees and SVMs require less computing power. The Google Colab platform, which makes free GPUs available for research, was used for the studies in order to solve this restriction. But there were drawbacks to this strategy as follows:

- Maximum runtime limitations in Google Colab were a big challenge, since it disconnects sessions on lengthy training cycles.
- The runtime issues faced: frequent memory-related or runtime errors that we faced during the model training, specifically when managing big batch sizes.
- Also, the limited persistent storage of Colab: since we frequently faced resets of sessions because of the repeated data uploads and model reinitializations.

B. Dataset Availability and Relevance

This is an important part of our framework: the availability of high-quality data that simulate real-world traffic scenarios. In this work, we used a dataset that ensures compatibility with the proposed ensemble mechanism and reflects typical WSN network conditions. While we selected a highly relevant dataset, it showed the following challenges:

- Constraints related to the limited availability of data required additional preprocessing and domain interpretation.
- Overlap with current studies: since there are many studies that utilized the same dataset in the literature in the context of the study domain.
- Infeasibility of data collection: As a university student, the collection of data that simulate real-world WSN traffic was not practical due to resource, time, and equipment limitations.

C. Model Integration and Quantization Trade-off

The integration of ML models into a single ensemble demands careful consideration of how they contribute to the final result. On the other hand, the quantization sensitivity factor, since the quantization technique may lead to a degradation in performance, convergence stability, especially when using batch normalization and dropout layers.

D. Generalization and Real-World Applicability

The final model showed significant findings in detecting intrusions. However, deploying such a system in a real-world WSN environment would face additional challenges, including:

- Real-time detection under limited hardware.
- Handling of imbalanced classes or rare attack types.
- Adaptability to unseen threats.

These factors draw attention to the discrepancy between simulation-based assessment and real-world implementation, which may be filled in subsequent research.

5.3 Future Work

The present evaluation is dataset-based (NSL-KDD and UNSW-NB15) and focuses on per-inference performance and energy estimates. To evaluate behavior under larger or more diverse WSN topologies (e.g., many nodes, multi-hop routing, heterogeneous traffic mixes), future work should (i) simulate the IDS at the network level (ns-3, OMNeT++) to measure end-to-end detection latency and communication overhead, or (ii) run hardware-in-the-loop experiments that deploy small classifiers on representative sensor nodes and the quantized DNN on the gateway. We expect per-inference computational cost to remain similar, but system-level factors (communication patterns, aggregation strategies, and class imbalance) will influence false-positive impact and aggregated energy consumption. Security validation should include two operational measurements in addition to accuracy:

- Detection latency: measure the elapsed time between packet arrival (or detection-window closure) and alert issuance, including preprocessing, inference, and any serialization/communication overhead. Report the distribution (median and 95th percentile) of end-to-end detection latency across representative traffic.
- False-positive energy cost: quantify the average extra communication and processing triggered per false alarm (for example, uplink alert transmission, logging, or gateway-side heavy analysis). Multiply the average extra operations or bytes transmitted per false alarm by per-operation or per-byte energy to estimate the additional energy burden that false positives impose.

Including these metrics provides a more complete operational cost analysis and helps select models that minimize the combined cost of misclassification and resource consumption.

Future recommendations include the following:

- This work can be extended by expanding the domain of optimization by including other techniques such as pruning and knowledge distillation.
- Employing more benchmark datasets in real-time scenarios.
- Improve detection by exploring more complex DL architectures, such as CNNs, LSTMs, or attention-based networks.

References

- Abduvaliyev, A., Lee, S., & Lee, Y.-K. (2010). Energy efficient hybrid intrusion detection system for wireless sensor networks. In *2010 International Conference on Electronics and Information Engineering* (pp. V2-25–V2-29). IEEE. https://doi.org/10.1109/ICEIE.2010.5559708
- Acharya, R. Y., Jeune, L. L., Mentens, N., Ganji, F., & Forte, D. (2023). Quantization-aware neural architectural search for intrusion detection. *arXiv*. https://doi.org/10.48550/arXiv.2311.04194
- Ahmed, U., Nazir, M., Sarwar, A., Ali, T., Aggoune, E.-H. M., Shahzad, T., & Khan, M. A. (2025). Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering. *Scientific Reports*, *15*(1), 1726. https://doi.org/10.1038/s41598-025-85866-7
- Alotaibi, Y., & Ilyas, M. (2023). Ensemble-learning framework for intrusion detection to enhance Internet of Things' devices security. *Sensors*, *23*(12), 5568. https://doi.org/10.3390/s23125568
- Bella, K., Guezzaz, A., Benkirane, S., Azrour, M., &Ennajar, S. (2025). Pruning and quantization of CNN-based intrusion detection systems for cyber-physical systems. In Y. Farhaoui et al. (Eds.), *Intersection of artificial intelligence, data science, and cutting-edge technologies: From concepts to applications in smart environment* (Vol. 1353, pp. 348–354). Springer. https://doi.org/10.1007/978-3-031-88304-0_48
- Charbuty, B., & Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, *2*(1), 20–28. https://doi.org/10.38094/jastt20165
- Chithra Rani, P. R., &Baalaji, K. (2025). Deep learning-based ensemble stacking for enhanced intrusion detection in IoT-edge platforms. *Discover Applied Sciences*, *7*(8), 928. https://doi.org/10.1007/s42452-025-06871-z
- Costa, V. G., & Pedreira, C. E. (2023). Recent advances in decision trees: An updated survey. *Artificial Intelligence Review*, *56*(5), 4765–4800. https://doi.org/10.1007/s10462-022-10275-5

- Devi, M., Nandal, P., &Sehrawat, H. (2025). Federated learning-enabled lightweight intrusion detection system for wireless sensor networks: A cybersecurity approach against DDoS attacks in smart city environments. *Intelligent Systems with Applications, 27*, 200553. https://doi.org/10.1016/j.iswa.2025.200553
- Filho, G. R. P., Villas, L. A., Freitas, H., Valejo, A., Guidoni, D. L., & Ueyama, J. (2018). ResiDI: Towards a smarter smart home system for decision-making using wireless sensors and actuators. *Computer Networks, 135*, 54–69. https://doi.org/10.1016/j.comnet.2018.02.009
- Gebremariam, G. G., Panda, J., & Indu, S. (2023). Design of advanced intrusion detection systems based on hybrid machine learning techniques in hierarchically wireless sensor networks. *Connection Science, 35*(1), 2246703. https://doi.org/10.1080/09540091.2023.2246703
- Gowdhaman, V., & Dhanapal, R. (2022). An intrusion detection system for wireless sensor networks using deep neural network. *Soft Computing, 26*(23), 13059–13067. https://doi.org/10.1007/s00500-021-06473-y
- Guo, W., He, M., Huang, C., He, H., Song, S., Zhang, J., &Letaief, K. B. (2024). Model-driven deep learning for distributed detection with binary quantization. *arXiv*. https://doi.org/10.48550/arXiv.2404.00309
- Han, S.-H., Kim, K. W., Kim, S., & Youn, Y. C. (2018). Artificial neural network: Understanding the basic concepts without mathematics. *Dementia and Neurocognitive Disorders, 17*(3), 83. https://doi.org/10.12779/dnd.2018.17.3.83
- Hernández, N., Almeida, F., & Blanco, V. (2024). Optimizing convolutional neural networks for IoT devices: Performance and energy efficiency of quantization techniques. *The Journal of Supercomputing, 80*(9), 12686–12705. https://doi.org/10.1007/s11227-024-05929-w
- Huanan, Z., Suping, X., &Jiannan, J. (2021). Security and application of wireless sensor network. *Procedia Computer Science, 183*, 486–492. https://doi.org/10.1016/j.procs.2021.02.088
- Kannan, B. B., & Srinivasan, S. (2023). Energy efficient intrusion detection system in wireless sensor networks. *International Journal of Computer Engineering and Technology, 14*(3). https://doi.org/10.34218/IJCET.14.03.002

- Karatas, G., Demir, O., &Sahingoz, O. K. (2018). Deep learning in intrusion detection systems. In *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)* (pp. 113–116). IEEE. https://doi.org/10.1109/IBIGDELFT.2018.8625278
- Karimi, H., Derr, T., & Tang, J. (2019). Characterizing the decision boundary of deep neural networks. *arXiv*. https://doi.org/10.48550/arXiv.1912.11460
- Karthikeyan, M., Manimegalai, D., &RajaGopal, K. (2024). Firefly algorithm based WSN-IoT security enhancement with machine learning for intrusion detection. *Scientific Reports*, *14*(1), 231. https://doi.org/10.1038/s41598-023-50554-x
- Khan, M. N., Rahman, H. U., Almaiah, M. A., Khan, M. Z., Khan, A., Raza, M., ... & Khan, R. (2020). Improving energy efficiency with content-based adaptive and dynamic scheduling in wireless sensor networks. Ieee Access, 8, 176495-176520.https://doi.org/10.1109/ACCESS.2020.3026939
- Kim, D., & Heo, T.-Y. (2022). Anomaly detection with feature extraction based on machine learning using hydraulic system IoT sensor data. *Sensors*, *22*(7), 2479. https://doi.org/10.3390/s22072479
- Kim, T., Vecchietti, L. F., Choi, K., Lee, S., & Har, D. (2020). Machine learning for advanced wireless sensor networks: A review. IEEE Sensors Journal, 21(11), 12379-12397.https://doi.org/10.1109/JSEN.2020.3035846
- Kozak, J., Probierz, B., Kania, K., &Juszczuk, P. (2022). Preference-driven classification measure. *Entropy*, *24*(4), 531. https://doi.org/10.3390/e24040531
- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539
- Li, G., Gao, W., & Gao, W. (2024). Quantization techniques. In *Point cloud compression* (pp. 97–112). Springer Nature Singapore. https://doi.org/10.1007/978-981-97-1957-0_5
- Lima, M. G., Carvalho, A., Álvares, J. G., das Chagas, C. E., & Goldschmidt, R. R. (2024). Impacts of data preprocessing and hyperparameter optimization on the performance of machine learning models applied to intrusion detection systems. *arXiv*. https://doi.org/10.48550/arXiv.2407.11105
- Madhuri, D. K. (2022). A new level intrusion detection system for node level drop attacks in wireless sensor network. *Journal of Algebraic Statistics*, *13*(1).

- Manoharan, J. (2021). A novel user layer cloud security model based on chaotic Arnold transformation using fingerprint biometric traits. *Journal of Innovative Image Processing*, *3*(1), 36–51. https://doi.org/10.36548/jiip.2021.1.004
- Meng, W., Li, W., Su, C., Zhou, J., & Lu, R. (2018). Enhancing trust management for wireless intrusion detection via traffic sampling in the era of big data. *IEEE Access*, *6*, 7234–7243. https://doi.org/10.1109/ACCESS.2017.2772294
- Mohy-Eddine, M., Guezzaz, A., Benkirane, S., Azrour, M., &Farhaoui, Y. (2023). An ensemble learning based intrusion detection model for industrial IoT security. *Big Data Mining and Analytics*, *6*(3), 273–287. https://doi.org/10.26599/BDMA.2022.9020032
- Moustafa, N., & Slay, J. (2016). The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Information Security Journal: A Global Perspective*, *25*(1–3), 18–31. https://doi.org/10.1080/19393555.2015.1125974
- Msolli, A., Helali, A., &Maaref, H. (2018). New security approach in real-time wireless multimedia sensor networks. *Computers & Electrical Engineering*, *72*, 910–925. https://doi.org/10.1016/j.compeleceng.2018.01.016
- Mustafa, R., Sarkar, N., Mohaghegh, M., Pervez, S., & Vohra, O. (2025). Cross-layer analysis of machine learning models for secure and energy-efficient IoT networks. *Sensors*, *25*(12), 3720. https://doi.org/10.3390/s25123720
- Naidu, G., Zuva, T., & Sibanda, E. M. (2023). A review of evaluation metrics in machine learning algorithms. In R. Silhavy & P. Silhavy (Eds.), *Artificial intelligence application in networks and systems* (Vol. 724, pp. 15–25). Springer. https://doi.org/10.1007/978-3-031-35314-7_2
- Perumal, G., Subburayalu, G., Abbas, Q., Naqi, S. M., & Qureshi, I. (2023). VBQ-Net: A novel vectorization-based boost quantized network model for maximizing the security level of IoT system to prevent intrusions. *Systems*, *11*(8), 436. https://doi.org/10.3390/systems11080436
- Putrada, A. G., Alamsyah, N., Fauzan, M. N., Prabowo, S., &Oktaviani, I. D. (2024). QUIDS: A novel edge-based botnet detection with quantization for IoT device pairing. *Indonesia Journal on Computing (Indo-JC)*, 29–41. https://doi.org/10.34818/INDOJC.2023.8.3.878

- Rekha, K. S., Sreenivas, T. H., & Kulkarni, A. D. (2018). Remote monitoring and reconfiguration of environment and structural health using wireless sensor networks. *Materials Today: Proceedings*, *5*(1), 1169–1175. https://doi.org/10.1016/j.matpr.2017.11.198
- Rojas, R. (1996). The backpropagation algorithm. In R. Rojas (Ed.), *Neural networks* (pp. 149–182). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-61068-4_7
- Rose, J. R., Swann, M., Bendiab, G., Shiaeles, S., & Kolokotronis, N. (2021). Intrusion detection using network traffic profiling and machine learning for IoT. In *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)* (pp. 409–415). IEEE. https://doi.org/10.1109/NetSoft51509.2021.9492685
- Rose, T., Kifayat, K., Abbas, S., & Asim, M. (2020). A hybrid anomaly-based intrusion detection system to improve time complexity in the Internet of Energy environment. *Journal of Parallel and Distributed Computing*, *145*, 124–139. https://doi.org/10.1016/j.jpdc.2020.06.012
- Roy, B., & Cheung, H. (2018). A deep learning approach for intrusion detection in Internet of Things using bi-directional long short-term memory recurrent neural network. In *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)* (pp. 1–6). IEEE. https://doi.org/10.1109/ATNAC.2018.8615294
- S. S., & Joe, V. (2021). Metric routing protocol for detecting untrustworthy nodes for packet transmission. *Journal of Information Technology and Digital World*, *3*(2), 67–76. https://doi.org/10.36548/jitdw.2021.2.001
- Sakthimohan, M., Deny, J., & Rani, G. E. (2024). Secure deep learning-based energy efficient routing with intrusion detection system for wireless sensor networks. *Journal of Intelligent & Fuzzy Systems*, *46*(4), 8587–8603. https://doi.org/10.3233/JIFS-235512
- Seba, A., Nouali-Taboudjemat, N., Badache, N., & Seba, H. (2019). A review on security challenges of wireless communications in disaster emergency response and crisis management situations. *Journal of Network and Computer Applications*, *126*, 150–161. https://doi.org/10.1016/j.jnca.2018.11.010
- Sharmila, B. S., &Nagapadma, R. (2023). Quantized autoencoder (QAE) intrusion detection system for anomaly detection in resource-constrained IoT devices using RT-IoT2022 dataset. *Cybersecurity*, *6*(1), 41. https://doi.org/10.1186/s42400-023-00178-5

- Shen, J., Yang, W., Chu, Z., Fan, J., Niyato, D., & Lam, K.-Y. (2024). Effective intrusion detection in heterogeneous Internet-of-Things networks via ensemble knowledge distillation-based federated learning. In *ICC 2024 - IEEE International Conference on Communications* (pp. 2034–2039). IEEE. https://doi.org/10.1109/ICC51166.2024.10622262
- Shmilovici, A. (2005). Support vector machines. In O. Maimon & L. Rokach (Eds.), *Data mining and knowledge discovery handbook* (pp. 257–276). Springer-Verlag. https://doi.org/10.1007/0-387-25465-X_12
- Smys, S., Basar, A., & Wang, H. (2020). Hybrid intrusion detection system for Internet of Things (IoT). *Journal of ISMAC, 2*(4), 190–199. https://doi.org/10.36548/jismac.2020.4.002
- Sowmya, T., & Mary Anita, E. A. (2023). A comprehensive review of AI based intrusion detection system. *Measurement: Sensors*, *28*, 100827. https://doi.org/10.1016/j.measen.2023.100827
- Staudemeyer, R. C. (2012). The importance of time: Modelling network intrusions with long short-term memory recurrent neural networks (Doctoral dissertation, UNIVERSITY OF THE WESTERN CAPE).
- Sundaramoorthy, K., Purushothaman, K. E., Sonia, J. J., & Kanthimathi, N. (2024). Enhancing cybersecurity in cloud computing and WSNs: A hybrid IDS approach. *Computers & Security*, *147*, 104081. https://doi.org/10.1016/j.cose.2024.104081
- Talukder, M. A., Sharmin, S., Uddin, M. A., Islam, M. M., & Aryal, S. (2024). MLSTL-WSN: Machine learning-based intrusion detection using SMOTETomek in WSNs. *International Journal of Information Security, 23*(3), 2139–2158. https://doi.org/10.1007/s10207-024-00833-z
- Tampubolon, J., &Kusrini, K. (2025). Comparative performance of LSTM and DNN in sentiment analysis. JurnalMandiri IT, 14(1), 1-11.https://doi.org/10.35335/mandiri.v13i4.403
- Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications* (pp. 1–6). IEEE. https://doi.org/10.1109/CISDA.2009.5356528

- Ullah, I., & Mahmoud, Q. H. (2021). Design and development of a deep learning-based model for anomaly detection in IoT networks. *IEEE Access, 9*, 103906–103926. https://doi.org/10.1109/ACCESS.2021.3094024
- Umar, H. G. A., Yasmeen, I., Aoun, M., Mazhar, T., Khan, M. A., Jaghdam, I. H., & Hamam, H. (2025). Energy-efficient deep learning-based intrusion detection system for edge computing: A novel DNN-KDQ model. *Journal of Cloud Computing*, *14*(1), 32. https://doi.org/10.1186/s13677-025-00762-9
- Vuckovic, A., Radivojevic, V., Chen, A. C. N., & Popovic, D. (2002). Automatic recognition of alertness and drowsiness from EEG by an artificial neural network. *Medical Engineering & Physics*, *24*(5), 349–360. https://doi.org/10.1016/S1350-4533(02)00030-9
- Wang, Z., Chen, H., Yang, S., Luo, X., Li, D., & Wang, J. (2023). A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization. *PeerJ Computer Science, 9*, e1569. https://doi.org/10.7717/peerj-cs.1569
- Wei, L., Ma, Z., Yang, C., & Yao, Q. (2024). Advances in the neural network quantization: A comprehensive review. *Applied Sciences*, *14*(17), 7445. https://doi.org/10.3390/app14177445
- Xu, Z., et al. (2025). Deep learning-based intrusion detection systems: A survey. *arXiv*. https://doi.org/10.48550/arXiv.2504.07839
- Yagiz, M. A., &Goktas, P. (2025). LENS-XAI: Redefining lightweight and explainable network security through knowledge distillation and variational autoencoders for scalable intrusion detection in cybersecurity. *arXiv*. https://doi.org/10.48550/arXiv.2501.00790

تطوير نظام كشف تسلل هجين موفّر للطاقة لشبكات الاستشعار اللاسلكية

إعداد : مراد محمد فؤاد جمل

المشرف : د. رائد الزغل

الملخص

تُستخدم شبكات الاستشعار اللاسلكية (WSNs) بشكل متزايد في العديد من التطبيقات الواقعية، بما في ذلك المدن الذكية، والرعاية الصحية، والأنظمة الصناعية؛ إلا أن محدودية مواردها الحاسوبية تجعلها أكثر عرضة للهجمات السيبرانية. وقد أظهرت أنظمة كشف التسلل (IDS) المعتمدة على خوارزميات الذكاء الاصطناعي (AI)، بما في ذلك خوارزميات التعلم العميق (DL) والتعلم الآلي(ML) ، قدرتها على اكتشاف أنماط التهديدات المعقدة. ومع ذلك، فإن متطلبات المعالجة العالية لهذه الخوارزميات تحدّ من إمكانية نشرها في بيئات شبكات الاستشعار اللاسلكية ذات الموارد المحدودة.

تعالج هذه الدراسة المفاضلة بين دقة الكشف والتعقيد الحاسوبي، وتقدم نموذج تعلم عميق موفّرًا للطاقة لكشف التسلل في شبكات الاستشعار اللاسلكية، بهدف اكتشاف أحدث التهديدات السيبرانية مع الحفاظ على دقة عالية وتكلفة حسابية منخفضة. تم استخدام أسلوب تحسين التكميم (Quantization) على نموذج الشبكة العصبية (NN) للحد من استهلاك الطاقة. وأظهرت نتائج الدراسة أن النموذج المكمَّم يقلل بشكل ملحوظ من التكلفة الحاسوبية واستخدام الذاكرة مع الحفاظ على أداء كشف جيد.

تم إجراء التنبؤ النهائي باستخدام أسلوب التجميع (Ensemble) من خلال دمج ثلاثة نماذج هي: شجرة القرار(DT) ، وآلة المتجهات الداعمة(SVM) ، والشبكة العصبية المكمَّمة. وأظهرت النتائج النهائية أن نموذج التجميع حقق دقة بلغت 0.9238 بزمن تنبؤ قدره 4.1391 ثانية، مما يبرز توازنًا قويًا بين فعالية الكشف والكفاءة الحاسوبية. بالإضافة إلى ذلك، حقق نموذج التجميع قيمًا مرتفعة لكل من الدقة (Precision)بمقدار 0.9965، والاستدعاء (Recall) بمقدار 0.9232، ودرجة F1 بمقدار 0.9585، مما يشير إلى كشف موثوق لكل من الحركة الطبيعية والخبيثة. وبالمقارنة مع النماذج الفردية والتقييمات على مجموعة بياناتNSL−KDD ، أظهر النهج المقترح متانة وتعميمًا أفضل مع الحفاظ على حجم ذاكرة صغير مناسب لبيئات شبكات الاستشعار اللاسلكية ذات الموارد المحدودة.

الكلمات المفتاحية : الذكاء الاصطناعي(AI) ، التعلم العميق(DL) ، التعلم الآلي(ML) ، شبكات الاستشعار اللاسلكية(WSNs) ، نظام كشف التسلل(IDS) ، التكميم.(Quantization)