HYBRID AI-POWERED INTRUSION DETECTION SYSTEM FOR WIRELESS NETWORKS

BY

ONYIBO CHUKWUEBUKA GIDEON
GOU/U22/CSC/864

DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS, FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY,
GODFREY OKOYE UNIVERSITY, ENUGU STATE. **IN PARTIAL FULFILLMENT OF THE AWARD OF BACHELOR OF SCIENCE (B.SC), DEGREE IN COMPUTER SCIENCE.**

SUPERVISOR: DR. FRANK OKEBANAMA

JUNE, 2026.

APPROVAL PAGE

This research, titled "HYBRID AI-POWERED INTRUSION DETECTION SYSTEM FOR WIRELESS NETWORKS," has been assessed and approved by the Department of Computer Science and Mathematics, Faculty of Computing and Information Technology, Godfrey Okoye University, Enugu State, Nigeria.

....

Supervisor Signature/Date

Dr. Frank Okebanama

....

External Examiner Signature/Date

....

HOD, Department of Computer Science and Mathematics Signature/Date

Dr. Frank Okebanama

....

Dean, Faculty of Computing and Information Technology Signature/Date

Prof. I. A. Ajah

CERTIFICATION

I certify that I completed the research included herein and that it was primarily based on my own observations and investigations. Where other sources are used, due acknowledgement is given. I accept full responsibility for the authenticity of this work.

_____

ONYIBO CHUKWUEBUKA GIDEON

GOU/U22/CSC/864

DEDICATION

This project is dedicated first and foremost to God Almighty, in whose hands all things are possible. To my parents and family, who sacrificed so much to see me reach this point — your love and support are the foundation upon which this work was built. To every student who dares to pursue knowledge beyond the ordinary, may this work inspire you.

ACKNOWLEDGEMENTS

All glory and honor to God Almighty for granting me the wisdom, health, and perseverance to finish this research. My faith in him has been my strength along the way.

The supervision by my supervisor, Dr. Frank Okebanama, from its inception to its submission has been a great inspiration; his patience, expertise, and dedication are much appreciated. His careful feedback and academic guidance were priceless, and I am lucky to have had a dedicated supervisor.

I must express my gratitude to all the teaching staff of the Department of Computer Science and Mathematics, Godfrey Okoye University, for the good education they rendered to me and the passion for computing they gave me.

To my parents, I can't say how grateful I am. Every page of this work reflects your prayers and sacrifices and your unconditional love. This accomplishment is mine as well. I am particularly grateful to my siblings. I can't thank you enough for your encouragement, understanding, and steadfast support during the toughest times of this research. You were there to support me through the times of doubt and made me feel confident to move forward. I am blessed to have you in my life.

Thank you to all my friends and course mates for all the hardships, humour, and encouragement over the course of this programme. This trip was an unforgettable one.

ABSTRACT

Conventional signature-based intrusion detection systems (IDSs) are not very effective at detecting subtle network-level attacks such as deauthentication, disassociation, and rogue access point impersonation in IEEE 802.11-based wireless networks. This research has devised and developed a Hybrid AI-Powered Intrusion Detection System (IDS) for wireless networks, consisting of two models: a supervised Random Forest model for known attack classification and an unsupervised Isolation Forest model for zero-day anomaly detection. It has been combined with a rule-based expert system and a SHAP (SHapley Additive exPlanations) explainability engine to provide structured explanations in plain English of the nature of each detected intrusion, and is presented via a Flask-based near-real-time Web Dashboard. The system was trained on the AWID3 wireless benchmark dataset with a 33/41/40 classification breakdown (Deauth, Disas, RogueAP) using a file-level train-test split (205730/313193 files), with class distribution matching. On the unseen data for testing, the Random Forest classifier had an accuracy of 99.44%, precision of 99.55%, recall of 99.44%, and F1 score of 99.47%. It successfully recalled 1.00% of instances from each of the three attack categories.. To test the generalization of the wireless network datasets beyond the controlled laboratory environment, five synthetic wireless network datasets were created from the statistical distribution of the AWID3 training datasets, and the accuracy was found to be 97.87% on average. The results indicate that the proposed hybrid architecture is capable of solving the three identified research challenges that wireless IDS research lacks: a lack of wireless-specific hybrid IDS implementation, a lack of dual supervised and unsupervised detection architectures in IEEE 802.11 environments, and the universal absence of explainability mechanisms in wireless IDS research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

Wireless networks are an integral part of the communication infrastructure and serve an array of applications, including personal mobile wireless, enterprise communications, industry wireless, and Internet of Things (IoT). They are based on the IEEE 802.11 family of standards and provide unprecedented flexibility, mobility, and cost efficiency, allowing billions of devices to communicate without the limitations of physical infrastructure (Saranya et al., 2020). The explosive growth of wireless technology has created the expectation for seamless connectivity. At the same time, it is one of the most vulnerable networking technologies from a security standpoint.

Unlike in wired networks, physical proximity to the wires or hardware is not necessary in order for one to intercept the communication in wireless networks because the communication signals travel through the air and can be accessed by anyone within the range using a packet sniffer or wireless protocol analyzer (Ozkan-Okay et al., 2021). This is a basic property that makes wireless networks susceptible to various attacks on the IEEE 802.11 protocol. Some of the most common and harmful of these attacks are deauthentication attacks, which involve an attacker sending forged IEEE 802.11 management frames to disrupt legitimate users from accessing the wireless points; disassociation attacks, which make use of unencrypted IEEE 802.11 management frames to deliberately cause the breaking down of the connection between a wireless client and an access point; and rogue access point spoofing, also referred to as evil twin attacks, where an attacker sets up a fraudulent access point to intercept sensitive information and credentials of the users who connect to it.

The history of wireless security holes demonstrates that wireless security threats are never going away and are always changing shape. However, basic flaws in the encryption design of the early Wired Equivalent Privacy (WEP) standard were soon exploited, and within a few years of its adoption, the standard was broken. WPA and WPA2 were improvements on their predecessors, but these were also discovered to be susceptible to various attacks, including KRACK (Key Reinstallation Attack), which was able to exploit weaknesses in the WPA2 4-way handshake to allow traffic to be decrypted. These developments confirm that encryption is not enough to ensure the security of wireless networks and that an active, intelligent monitoring mechanism is necessary to achieve good security (Ozkan-Okay et al., 2021).

Traditional Intrusion Detection Systems (IDS) constitute the main defensive measure against threats to wireless networks. For signature-based systems (Snort and Suricata), they are used to look for known attack patterns in network traffic, whereas for wireless systems (Kismet), they are used to detect IEEE 802.11 frames with known threat signatures (Acka et al., 2025). Both of the following approaches have serious drawbacks in wireless environments. Signature-based systems are always reactive, and they can never catch zero-day attacks or changing signature forms of known threats. However, the problem with anomaly-based systems is that the high false positive rates are a frequent occurrence in wireless environments where legitimate changes in signals due to device mobility, interference, and network congestion are often mistaken for intrusions, thus leading to alert fatigue among the network administrators (Vinayakumar et al., 2019).

The most promising answer to these drawbacks is artificial intelligence, especially machine learning. Algorithms in Machine Learning can learn complex patterns from large amounts of historical wireless traffic data and accurately classify network behavior with consistently high accuracy (Yellepeddi et al., 2024). Random Forest (RF) ensemble classifiers are found to be highly

effective in wireless IDS tasks with maximum accuracy beyond 99% on the AWID benchmark dataset (Saranya et al., 2020; Ikeri, 2025). In addition to supervised detection, unsupervised approaches like Isolation Forest can enhance the detection capability of the system by detecting instances that deviate structurally from the normal distribution, without needing labelled training samples (Acka et al., 2025).

In addition to the accuracy of detection, explainability has become another crucial demand for AI-based security systems. Most current machine learning-based IDSs are black boxes and simply return classification results, without providing any explanation of why a specific traffic instance is malicious. This opacity makes them impractical to use for network administrators who need to understand, verify, and take action on alerts quickly (Masengo Wa Umba et al., 2022). This work directly tackles this issue by incorporating a rule-based expert system with SHAP (SHapley Additive exPlanations) analysis to produce threat explanations and response recommendations in plain language for all intrusions.

1.2 Statement of the Problem

Despite the broad implementation of traditional security methods, applications still present a high vulnerability risk on the protocol level. IEEE 802.11 communication is open, and adversaries can listen, capture, and access the network traffic without being physically on the network, as described by Ozkan-Okay et al., 2021.

Current Wireless Intrusion Detection Systems (WIDS) are mainly based on signature-based detection, which needs regular manual updates and cannot detect new or changing attack variants. When the attack is wireless-specific, such as a deauthentication attack, disassociation attack, and rogue access point impersonation, the general-purpose IDS tools are not able to detect these attacks

without adaptation for the wireless environment (Ozkan-Okay et al., 2021; Sowmya et al., 2023). Moreover, AI-enabled IDS systems, which have proven to be successful in detection precision, work as “black boxes,” providing no explanations about how their detection mechanisms operate, the reasons for setting off an alarm, or the required remedial actions (Acka et al., 2025; Yellepeddi et al., 2024).

1.3 Aim and Objectives of the Study

The aim is to design and implement a hybrid AI-powered intrusion detection system for wireless networks. The specific objectives are to:

1. Identify intrusion risks on wireless networks
2. Design a hybrid intrusion detection approach that monitors access point impersonation attacks and others.
3. Embed a rule-based expert system with SHAP (SHapley Additive exPlanations) analysis.
4. Develop a web application that uses Flask to present an emulated near-real-time monitoring dashboard.
5. Compare the performance of both models according to classical metrics.

1.4 Significance of the Study

This study has some notable contributions to the areas of wireless network security, machine learning, and explainable AI. The following contributions are described:

- The study proves the feasibility of the dual-model AI-based detection architecture in a simulation-based academic setting by leveraging the complementary strengths of both supervised and unsupervised learning for more extensive coverage of threats compared to

the existing wireless IDS literature, which is dominated by single-model approaches (Sowmya et al., 2023).

- By using only the AWID3 dataset, the most widely known benchmark for wireless-specific IEEE 802.11 IDS, every part of the system is based on real wireless network behavior, and results can be compared to previous studies on the AWID dataset, such as Abdulhammed et al. (2018), who obtained 99.64% accuracy by applying Random Forest on the AWID dataset (Ozkan-Okay et al., 2021).
- The black-box opacity identified as one of the major obstacles to the practical implementation of AI-based IDS in operational environments (Acka et al., 2025; Yellepeddi et al., 2024) is addressed by incorporating a rule-based expert system, which helps network administrators to receive transparent, actionable, and auditable threat explanations.
- The near real-time streaming dashboard takes the project beyond a static analysis script and provides a useful and workable prototype, thereby offering a repeatable framework for future research and application of AI-based IDS in a wireless network security environment.

1.5 Scope of the Study

This project will focus on designing and implementing an AI-based intrusion detection system for wireless networks using a dataset. The system is designed to be independent of actual wireless network traffic data collected through live hardware-based packet capture, physical deployment on wireless infrastructure, and real-time network response enforcement – only pre-captured wireless network traffic data from the AWID3-CSV dataset provided in CSV format is used. This study does not cover "true real-time" monitoring and automated attack prevention.

The system aims at three types of IEEE 802.11 wireless network attacks: De-authentication attack, Disassociation attack, Rogue access point impersonation attack. Machine learning can only be used with two algorithms: supervised multi-class detection with the Random Forest classifier, and unsupervised anomaly detection with the Isolation Forest. The explainability is provided via a rule-based expert system and SHAP analysis, and is fully implemented in the local system environment.

1.6 Limitations of the Study

This study recognizes the following constraints that should be taken into account when interpreting the results of this study. Firstly, the system relies completely on the AWID3 data; this means its ability to detect is constrained by the quality, diversity, and representativeness of the AWID3 data. Patterns that are not seen in AWID3 cannot be detected, and trained models might need to be retrained to generalize to substantially different wireless networks or newer IEEE 802.11 protocols. Second, the near-real-time processing design simulated does not take into account all the complexities of real wireless environments, such as dynamic channel conditions, simultaneous multi-user traffic, and real-time device mobility patterns.

Thirdly, the Isolation Forest can produce false positive traffic distributions that are meaningfully different from its training base. Fourth, AWID3 is a controlled lab dataset, and the high accuracy of classification is representative of the uniformity of patterns of attacks recorded in the laboratory and not representative of the diversity of actual deployments. The limitations are overcome by careful pre-processing, careful evaluation, and clear reporting, but are significant if this effort is extended to live deployment**.**

1.7 Operational Definition of Terms

Artificial Intelligence (AI): A simulation of cognitive functions of humans, such as learning pattern recognition, classification, and decision support by computer systems.

Intrusive Detection System (IDS): A device that is used to detect intrusions in a network or system and alert the administrator to suspicious activity or breaches of security.

Wireless Network: An IEEE 802.11 or other wireless network that transmits data via radio frequency waves without the need for cable connections.

Machine Learning (ML): Artificial Intelligence (AI) technology subfield in which a computer system learns patterns from data and becomes better suited to a set of specific tasks without the need to be explicitly programmed for every situation.

Random Forest: An ensemble machine learning classifier that creates multiple decision trees during training and then outputs the class with the maximum number of votes; it does not suffer from overfitting and works excellently with unbalanced data.

Isolation Forest: An unsupervised anomaly detection algorithm that uses randomly selected feature splits to isolate observations and works on the idea that anomalies are structurally different from normal observations and thus are easier to isolate.

Expert System: A rule-based AI module that utilizes pre-established condition-action rules to provide context to the threat description, impact, and recommended actions for each threat category that is detected.

SHAP (SHapley Additive exPlanations): The explainable approach based on game theory that assesses the contribution of each feature to a decision provided by the model, offering explanations regarding the importance of features for making a prediction**.**

Explainable AI (XAI): The field of study in AI concerned with creating methods and techniques that allow AI model decisions to be transparent, interpretable, and understandable for human users.

Deauthentication Attack: An 802.11 attack that involves the adversary broadcasting IEEE 802.11 deauthentication management frames that cause legitimate users to disconnect from a wireless access point.

Disassociation Attack: An IEEE 802.11 management frame wireless attack that is used to attempt to break the association between a wireless client and its access point, thus excluding the client from the network.

Rogue Access Point Impersonation: An attack against a wireless network where an imposter network is created, mimicking a valid network, leading users to log in and subsequently revealing sensitive data and authentication details. Also known as the evil twin attack

The Aegean Wi-Fi Intrusion Dataset version 3 (AWID3): AWID3 is a publicly available real wireless network traffic benchmark dataset developed for training and testing wireless intrusion detection systems (WIDS) that contains IEEE 802.11 wireless network traffic with real labels..

Hybrid IDS: The IDS used is a hybrid IDS where two or more detection strategies are used together in the project, one supervised, which is a Random Forest Classifier, and one unsupervised, which is an Isolation Forest.

Simulated Real-Time Processing: An operational mode where the network traffic records from a pre-cut or "canned" collection are processed sequentially as if being collected live, to give the

appearance and behavior of real-time network monitoring without requiring live collection hardware.

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

The literature survey in this chapter is a thorough survey of the existing literature related to the design and implementation of an AI-based Intrusion Detection System for wireless networks. The review brings together important theoretical concepts, data sets, methodologies, and empirical results of studies from 2017 to 2025. This chapter aims to provide a solid background for the proposed system, discuss the work done by previous researchers on the topic of wireless network intrusion detection, discuss the limitations and gaps of the previous researchers, and justify the novelty of this research.

This chapter elaborates upon the vulnerabilities highlighted in Chapter One (deauthentication attack, disassociation attack, rogue access point impersonation in IEEE 802.11 wireless environments) and the shortcomings of traditional rule-based intrusion detection systems (IDS) highlighted in the chapter, and how machine learning combined with ensemble methods, unsupervised anomaly detection, and explainable AI can overcome these shortcomings. The literature is presented as follows: First, the theoretical background (Section 2.2) focusing on the five core pillars of this project will be presented, followed by related empirical works (Section 2.3), a summary of key findings (Section 2.4), and a reflection on the research gap addressed by this project (Section 2.5).

2.2 Theoretical Background

This section explains the fundamental theories, principles, and technical concepts at the heart of the proposed intrusion detection system (IDS) that is powered by artificial intelligence (AI). This

section provides an explanation of the key theories, principles, and technical concepts that are fundamental to the proposed AI-based intrusion detection system (IDS). The five key areas of knowledge covered are: Wireless network security and vulnerabilities; Intrusion detection system types and approaches; Machine learning for intrusion detection; Unsupervised anomaly detection (with emphasis on Isolation Forest); Explainable AI (SHAP and expert systems in cybersecurity).

2.2.1 Wireless Network Security & Vulnerabilities

Wireless networks are the primary way for personal and enterprise communications to be connected, based on the IEEE 802.11 family of standards. However, they differ from their wired peers in that they have an open broadcast medium with inherent security challenges. Wireless signals are transmitted through the air and can be received by any device within radio range. It is possible to intercept, monitor, or interfere with wireless communications without entering into contact with network infrastructure (Ozkan-Okay et al., 2021). This aspect makes it a very challenging environment for network security, requiring dedicated detection methods.

There are structural weaknesses in handling management frames in the IEEE 802.11 protocol family that can still be exploited. In most implementations, management frames control device association, authentication, and network discovery and are transmitted without authentication or encryption, making for easy spoofing by an adversary (Acka et al., 2025). This study focuses on three types of attacks, which take advantage of these vulnerabilities. Deauthentication attacks are designed to send spoofed IEEE 802.11 deauthentication management frames to truncate the connection of the legitimate users from access points. Disassociation attacks also take advantage of the same unencrypted management frame vulnerability to cause a client to disconnect from the access point, thus preventing access to the network. Rogue access point impersonation (Wang et al., 2022): deployment of a fake access point set up to look like a

legitimate access point to allow traffic interception, credential stealing, and man-in-the-middle positioning.

This project's main dataset, the Aegean Wi-Fi Intrusion Dataset (AWID3), has been especially designed to overcome the lack of wireless-specific benchmarks in IDS research. AWID is a collection of real IEEE 802.11 traffic data with labels containing data from a variety of wireless attack types, including the attacks covered in this project, as described by Ozkan-Okay et al. (2021). It shares with the other wireless-specific features, such as frame-level attributes not found in general network datasets, the greatest advantage of being the best choice for a benchmark for evaluation of systems specifically designed for IEEE 802.11 environments.

2.2.2 Intrusion Detection Systems: Types and Approaches

The Intrusion Detection System (IDS) is a system used to identify intrusions, malfeasance, or policy violations in network traffic or system activities and alert on such incidents (Ikeri, 2025). IDS technologies can be broadly categorized by detection method into two main types. These include Signature-Based and Anomaly-based Detection.

Signature-based IDS is based on the comparison of seen traffic to a set of pre-defined attack signatures or rules. Snort and Suricata are common examples of this type of tool. Signature-based systems are very effective against known and well-defined attacks, but the system cannot detect zero-day attacks or attack variants that have not been added to the system's signature database, and is expected to be constantly updated manually to be effective (Ozkan-Okay et al., 2021). Anomaly-based IDS uses a statistical baseline of normal network activity as a way of identifying abnormal activity as a possible intrusion. This approach can, in theory, identify new threats without relying

on an attack signature. Nevertheless, anomaly-based systems have a higher False Positive Rate (FPR), especially in dynamic wireless environments (Vinayakumar et al., 2019).

IDS is further classified by its deployment location. The network-based IDS (NIDS), Host-based IDS (HIDS), and Wireless-specific IDS (WIDS) monitor traffic on network segments, individual devices, and IEEE 802.11 protocol-aware, respectively, and enable the detection of management frame-based attacks that general NIDS monitoring tools cannot detect (Ozkan-Okay et al., 2021). The proposed system is based on an NIDS (dataset-driven) approach, having wireless-specific feature evaluation and clearly falling in the WIDS category.

2.2.3 Machine Learning for Intrusion Detection: Random Forest

Historically, the intrusion detection problem has been solved by defining rules, but with the support of machine learning, it has become possible to configure the system to learn complex patterns from past traffic without requiring any rules. The supervised learning approach is more effective for detecting known categories of attacks, while the unsupervised approach is more effective in detecting new attacks which are not been seen before (Yellepeddi et al., 2024).

Random Forest is one of the most successful supervised learning algorithms for IDS in various datasets and network topologies. Random Forest is an ensemble machine learning algorithm, in which several decision trees are built during training time; however, each of these decision trees is trained on randomly selected data sets and features. The output is decided by the majority voting of all these decision trees, performs well on imbalanced class distribution, has an inherent ranking of feature importance, and is seen to achieve competitive accuracy without extensive hyperparameter tuning (Sowmya et al., 2023; Saranya et al., 2020).

Random Forest's suitability for wireless IDS has been well proven through experience. In a thorough survey of 72 papers, Sowmya et al. (2023) found that Random Forest outperforms single classifiers with an accuracy range of 95% – 99%. Saranya et al. (2020) report the result of Random Forest 99.65% on KDD-CUP. The most relevant to this project is the result of Abdulhammed et al., cited in Ozkan-Okay et al., (2021), which are 99.64% accuracy on the AWID dataset with 32 features, as the direct benchmark that the results of this project are compared. Ikeri (2025) also finds RF to be the most accurate binary classification model when compared with all other models in four datasets, with 99.9% accuracy.

2.2.4 Unsupervised Anomaly Detection: Isolation Forest

Supervised approaches, like Random Forest, are good at identifying a limited number of known attacks with labels, but they inherently struggle with detecting novel attacks or attacks with zero days that do not have labelled training examples. A major drawback of supervised anomaly detection is that it is unable to detect anomalies that do not follow a known attack pattern (Ikeri, 2025).

Isolation Forest is an unsupervised anomaly detection method that is especially suitable for high-dimensional data of network traffic. While both density-based and distance-based anomaly detectors rely on the idea that anomalies occur when instances lie outside the density of the normal instances, Isolation Forest is based on the premise that anomalies have structural differences from normal instances, and thus are easier to isolate by randomly partitioning instances. The algorithm creates a set of decision trees through recursive selection of random features and random split points. Fewer splits will separate the outliers, resulting in smaller average path lengths and higher outlier scores. This method has low computational complexity, scaling ability to large datasets, no need for labelled training data, and is robust to class imbalance (Acka et al., 2025).

Random Forest for known threat classification with unsupervised anomaly detection techniques is the most complete IDS design strategy, as explained by Ikeri (2025). In Wireless Body Area Networks, Pant et al. (2023) prove unsupervised anomaly detection's effectiveness with an F1-score of 0.96 and AUC of 0.98. The suggested dual model approach for the current project is directly applicable to the IEEE 802.11 wireless IDS scenario through the AWID3 dataset.

2.2.5 Explainable AI: Expert Systems and SHAP in Cybersecurity

One major challenge of introducing AI-based intrusion detection systems into real-world scenarios is the lack of explainability in machine learning model decisions. Many successful models are black boxes that simply classify an instance of traffic as malicious without explaining why the specific instance was classified as malicious. The inability to interpret this makes it difficult for network administrators to trust the logs, make effective use of incident response, and hold them accountable in security-related contexts (Yellepeddi et al., 2024; Acka et al., 2025).

XAI means Explainable AI, which is an approach used to make the process of how an AI works more understandable. The SHAP method uses concepts of cooperative game theory and calculates the contribution of one particular variable to the output by allocating the output fairly among all the input variables (Ikeri, 2025). This generates quantitative explanations above the detection output, giving answers to the question of why the model made its classification decision, and thus creating an explanation layer that is transparent and auditable.

In addition to SHAP, rule-based expert systems can be used to generate contextual, domain-specific threat narratives from numerical feature contributions. An expert system is used to capture cybersecurity practitioner knowledge in the form of condition-action rules and generate descriptions of the attacks that can be understood by humans, including attack mechanisms, potential impacts, and recommended actions to take in response. Explainability features and expert

system rules are combined to yield a two-layer explainability architecture that is directly applicable to operational network security (Amachaghi, 2025; Acka et al., 2025). This project takes a dual-layer approach to this by creating an integrated explanation engine, directly addressing the gap in interpretability as revealed in the literature reviewed.

2.3 Review of Related Works

This section presents a critical review of both empirical and theoretical works related to the proposed system, arranged into three sub-groups: reviews and surveys about AI-based IDS, empirical studies about ML and ensemble models for IDS, and empirical studies directly related to wireless-specific and explainable IDS.

2.3.1 Reviews and Surveys on AI-Based Intrusion Detection

In IEEE Access, Ozkan-Okay et al. (2021) provide a systematic literature review on IDS technologies, methodologies, datasets, and tools for wired, wireless, and IoT environments. The review shows that Abdulhammed et al. used a Random Forest (RF) Classifier to get 32 features that give 99.64% accuracy on the AWID dataset, which is the key benchmark for this project. This broad scope of the review constitutes a limitation of this study, as there is not enough room to delve into any particular category of wireless attack or to combine explainability mechanisms.

Sowmya et al. (2023) present a detailed taxonomy of AI-based intrusion detection systems (IDSs) in both cloud and IoT settings, encompassing multiple Artificial Intelligence approaches. The analysis of 72 papers reveals an accuracy of 95% to 99%, and ensemble techniques generally achieve the highest detection accuracy in multi-class attack scenarios compared to single classifiers. Lack of protocol attack interpretation mechanisms and lack of detection of wireless-specific attacks are recognized as serious issues, thereby making the approach followed in this project justified.

The researchers, Saranya et al. (2020), used the KDD-CUP dataset to conduct tests on machine learning algorithms such as LDA, CART, and Random Forest to assess their performance in using them in IDS implementation.. The highest accuracy obtained is 99.65% by Random Forest, which also shows the best performance when considering the class-imbalanced data, with a greater number of normal traffic instances than attack instances. This study has been limited to the KDD-CUP dataset, which does not contain features specific to the wireless environment; this is remedied in this proposed work by using only AWID3.

Using a WSN dataset that is unique for Denial-of-Service attack detection, Fares et al. (2025) test Support Vector Machine (SVM), K-Nearest Neighbour (KNN), and Decision Tree classifiers to evaluate their performance for intrusion detection in Wireless Sensor Networks (WSN). The KNN model had the best accuracy of 99.76%, which proves the efficiency of machine learning techniques for intrusion detection in a wireless sensor network. The study points out that classical ML methods are still competitive in wireless environments with limited resources, as they can be seen as an alternative view to ensemble-based methods and that the evaluation of datasets for use in wireless IDS research is important.

2.3.2 Empirical Studies on ML Models in IDS

A scalable IDS framework is proposed by Vinayakumar et al., (2019), which uses deep learning networks and has a high testing accuracy rate of 97% to 99%, on benchmarks such as KDDCup99, NSL-KDD, UNSW-NB15, WSN- The study does concede, however, that DNN architectures are expensive to compute and are not easily explained, and that ensemble methods may be a better option if resources are limited. This restriction highlights the case of using AWID3 and Random Forest in the present study.

Wang et al. (2022) implement the AI@NTDS system based on the MITRE ATT&CK framework on a Cowrie honeypot dataset with 99.20% overall accuracy with the LightGBM algorithm and an F1 score of 99.80% with the LightGBM algorithm. The findings of the study illustrate the usefulness of feature importance analysis for threat level assessment and set a standard for threat detection by artificial intelligence design. The fact that it only addresses SSH and Telnet-type threats and not attacks to wireless protocols is its drawback.

Yellepeddi et al. (2024) examine the practical effectiveness of IDS using AI, reporting that AI-based systems can achieve a high rate of accuracy of over 98% and much lower false positive rates than signature-based systems. The paper outlines the increasing role of XAI in IDS, highlighting the limitations of existing AI-based systems in lacking captioning capabilities for humans, which this project aims to rectify.

Ikeri (2025) evaluates the IDS using AI technologies with multi-datasets, including NSL-KDD, UNSW-NB15, CICIDS-2017, and ToN_IoT datasets by employing the classifiers Random Forest, SVM, MLP, and LSTM. Random Forest has the highest binary classification accuracy of all algorithms in all three datasets, CICIDS-2017 (99.9%), NSL-KDD (95%), and ToN_IoT (94.5%), which makes it the default baseline algorithm for intrusion detection. A major finding of the thesis is that a combination of supervised and unsupervised techniques is the most promising direction for future IDS research, which is directly anticipated by a dual-model design employed in this project.

2.3.3 Wireless-Specific and Explainable AI Studies

In their thorough review, Acka et al. (2025) introduce a detailed analysis of AI-driven IDPS in wireless communication security, covering supervised learning, deep learning, ensemble learning, and explainable AI in the wireless security domain. The study highlights the need for ensemble-

based solutions for reliable classification of wireless attacks and the two main hurdles to their practical deployment: resource constraints and interpretability. The study explicitly asks for the use of XAI techniques in future wireless IDPS research, which was directly implemented in this project by the expert system and explainability engine SHAP.

In an empirical study, Masengo Wa Umba et al. (2022) train three anomaly detectors, namely Decision Tree, Naïve Bayes, and Deep ANN, on the NSL-KDD dataset using an end-to-end feature engineering scheme to implement intrusion detection using AI in SDWSN. The proposed Decision Tree-based detector provides an accuracy of 0.9998 in binary classification and memory-efficient operation for resource-limited wireless devices. The shortcoming is that this dataset is based on NSL-KDD instead of a wireless dataset specific to a protocol, which this project will remedy through exclusive use of AWID3.

Amachaghi (2025) presents an AI-powered IDPS for Open RAN 5G wireless networks, highlighting the growing significance of explainable AI in wireless network security and explaining how feature-based explanations and automated responses for guidance boost the operational value of AI-driven IDS for security practitioners. This work offers theoretical support for the explainability aspect of this project and proves that integrating explanation engines in the architecture of an IDS with Artificial Intelligence is feasible.

In Wireless Body Area Networks, Pant et al. (2023) have proposed a lightweight IDS system employing an unsupervised Convolutional Autoencoder with an F1 score of 0.96 and an AUC of 0.98 to detect both known and zero-day attack patterns. The study confirms the need for hybrid detection architectures that are based on the combination of supervised classification and

unsupervised anomaly detection, and directly applies to the Isolation Forest component of this project.

Ghadami (2025) introduced a sophisticated IDS system for IoT networks based on game-theory GANs and a parallel CNN-LSTM model that attained a precision of 99.86% on NSL-KDD and 98.67% accuracy on UNSW-NB15. The study points out the class imbalance problem as one of the main challenges and shows that the class distribution matching approach used in the AWID3 dataset in this project can significantly enhance minority class detection, which further validates the significance of class distribution matching.

In the paper titled "The Double-Edge Sword of AI in Cyber Security: Protection and Potential Attack Paths" (2022), Dash et al. explore how AI can be used to defend against attacks and how it can be leveraged for more sophisticated attacks. The paper includes theoretical foundations for the integration of explainability using AI in IDS, highlighting the need for quick and actionable threat intelligence for security response.

2.4 Summary of Literature Review

Overall, the literature surveyed shows that machine learning, especially ensemble methods, have played a significant role in the efficacy of IDS. Random Forest has been found to be the more accurate model in a variety of datasets and network environments and unsupervised methods can be used to detect new threats that are not included in the supervised model. Table 2.1 presents the findings of the key works that were reviewed, including their methods, findings, limitations, and direct relevance to the current study.

Table 2.1: Summary of Related Works

Author(s) & Year	Network Type	Technique	Dataset	Key Findings	Limitation	Relevance to Study
Ozkan-Okay et al. (2021)	Wired/Wireless/IoT	Systematic Review	NSL-KDD, AWID	RF on AWID 99.64%; ML improves adaptability	Broad scope; limited wireless depth	AWID benchmark; wireless IDS framework
Saranya et al. (2020)	General (Fog/IoT/5G)	LDA, CART, RF	KDD-CUP	RF 99.65%; best for imbalanced data	No wireless-specific dataset	Justifies RF algorithm selection
Fare et al. (2025)	Wireless Sensor Networks	SVM, KNN, Decision Tree	WSN-specific	KNN 99.76% for DoS detection	WSN-specific; limited attack diversity	Confirms ML suitability for wireless IDS
Vinayakumar et al. (2019)	General/WSN	DNN	NSL-KDD, WSN-DS, CICIDS-2017	DNN 97-99%; WSN-DS wireless application	High complexity; no interpretability	Justifies choosing RF over deep learning
Sowmya et al. (2023)	Cloud/IoT	ML, DL, Ensemble	Multiple	RF best for multi-class; 95-99% accuracy	Limited wireless depth; no XAI	Algorithm justification; XAI gap
Wang et al. (2022)	Enterprise Networks	LightGBM, RF, K-NN	Honeypot (Cowrie)	99.20% accuracy; 99.80% F1-score	Not wireless-specific	AI threat detection benchmark
Yellepeddi et al. (2024)	Mixed Networks	DT, SVM, CNN, RNN	Multiple	AI 98%+; reduced false positives	No wireless evaluation; no XAI	XAI gap; supports ML use

Author(s) & Year	Network Type	Technique	Dataset	Key Findings	Limitation	Relevance to Study
Ikeri (2025)	IoT/Mixed	RF, SVM, MLP, LSTM	NSL-KDD, UNSW-NB15, CICIDS-2017, ToN_IoT	RF top performer; hybrid approaches recommended	No wireless-specific dataset	Confirms RF; supports hybrid architecture
Acka et al. (2025)	Wireless Networks	ML, DL, Ensemble, XAI	Multiple	Ensemble effective for wireless; XAI needed	No AWID evaluation; XAI not implemented	Core wireless reference; motivates XAI
Masengo Wa Umba et al. (2022)	WSN (SDWSN)	DT, Naive Bayes, ANN	NSL-KDD	DT 0.9998 accuracy; feature engineering effective	NSL-KDD not wireless-specific	Wireless AI-IDS methodology
Amachaghi (2025)	Open RAN/5G Wireless	ML-based IDPS, XAI	Simulated Data	XAI effective for threat explanation in wireless	5G focus; not IEEE 802.11 specific	Grounding for expert system and SHAP
Pant et al. (2023)	WBAN	Autoencoder (unsupervised)	Custom physiological	F1=0.96; AUC=0.98; zero-day detection	WBAN-specific; narrow scope	Supports Isolation Forest component
Ghadami (2025)	IoT	GAN, CNN-LSTM	NSL-KDD, UNSW-NB15	99.86% precision; class balancing critical	Deep learning complexity	Informs class distribution matching
Dash et al. (2022)	General Cybersecurity	AI-based IDS (Survey)	Multiple	AI enhances detection; actionable intelligence critical	Not wireless-specific	Motivates explainability integration

Author(s) & Year	Network Type	Technique	Dataset	Key Findings	Limitation	Relevance to Study
Vanhoef & Piessens (2017)	IEEE 802.11 Wireless	Protocol Analysis	WPA2 Networks	KRACK exposes WPA2 four-way handshake vulnerability	Single vulnerability focus	Motivates active wireless IDS beyond encryption

2.5 Research Gap

Although there are considerable efforts towards the development of intrusion detection systems based on AI technology, no study integrates a wireless-specific dataset, hybrid supervised and unsupervised detection architecture, and explainability engine in one system. This study fills this gap by implementing a hybrid Random Forest (RF) and Isolation Forest (IF) detection system reported only on the AWID3 wireless benchmark dataset, and a rule-based expert system and SHAP (SHapley Additive Explanations) explainability engine, which produces understandable threat explanations for each intrusion detected (Ozkan-Okay et al., 2021; Ikeri, 2025; Acka et al., 2025).

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.0 Introduction

In this chapter, the system analysis and design of the proposed wireless network hybrid AI intrusion detection system are presented. The chapter provides a structured blueprint of the system design, which links the theoretical basis developed in Chapters One and Two with the practical implementation described in Chapter Four. This paper assesses the weaknesses of the present-day wireless intrusion detection system, outlines the functional and non-functional requirements of the proposed system, highlights the design method adopted for the same, models the system through Unified Modelling Language (UML), and concludes with the architectural design of the whole system.

This suggested system is a web-based application developed using Python programming that enables a network administrator to import wireless network traffic data in CSV file format, use a double-layered machine learning algorithm consisting of a Random Forest classifier along with the Isolation Forest anomaly detection model, get clear English descriptions of potential threat(s) from a rule-based expert system and a SHAP analysis tool, and analyze the result(s) via a near-real time Flask Dashboard. The design is all software-generated and simulation-based, and only tested against the AWID3 wireless benchmark dataset.

3.1 Research Design and Methodology

3.1.1 Research Design

This study uses a quantitative experimental research design. It is quantitative, as it relies on numerical benchmarks (accuracy, precision, recall, and F1-score) derived from predictions on a

held-out test set of 313,193 wireless network traffic records. It is experimental, as the proposed hybrid IDS architecture is carefully designed, implemented, and evaluated under controlled baseline conditions with the AWID3 benchmark dataset. The results are compared to the existing benchmarks from the literature. This study does not include human subjects, surveys, or qualitative data collection. Data is taken from a publicly available laboratory-captured benchmark of wireless network traffic, called the AWID3 dataset, and the results are examined by means of computational experimentation.

3.1.2Machine Learning Research Methodology

This study used the Cross-Industry Standard Process for Data Mining (CRISP-DM) as a guide for the development of the machine learning. CRISP-DM offers a sequential, iterative process for planning and implementing projects involving data, from problem understanding to deployment, ensuring that every aspect of the machine learning pipeline is covered. Table 3.1 provides a mapping of each CRISP-DM phase to the activities that are addressed in this project and the chapters in this project.

3.1.3 Software Design Methodology

In this case, the software development approach that will be utilized for the project is called Object-Oriented Analysis and Design (OOAD). OOAD involves partitioning of the system into objects with data and behavior. This is an appropriate method to take in this project for three reasons. There are three reasons why this is an appropriate method to take in this project. First, the processing pipeline in the system is clearly defined and follows the logic of each of the discrete classes: DataLoader, Preprocessor, DetectionEngine, ExplanationEngine, DashboardController, and SessionStore. Secondly, modularity allows individual components to be upgraded or replaced without affecting any other components. Third, encapsulation simplifies the interfaces of the

different methods to be reusable and simple to use, while remaining clear and comprehensible, and the machine learning model inference and SHAP calculation are hidden. In the project, the OOAD design artefacts created are Use Case Diagram, Activity Diagram, Class Diagram, and System Architecture Diagram, which are presented in Section 3.4.

3.2 System Analysis

3.2.1 Analysis of the Existing System

The existing IDSs deployed in wireless network environments are mainly signature-based or rule-based IDSs. The most popular tools in this category are Suricata and Snort, both of which are network-based intrusion detection systems that scan packet headers or payloads in comparison to a database of attack rules. For wireless-specific scenarios, passive IEEE 802.11 frame monitoring and wireless security auditing tools like Kismet and Aircrack-ng (Acka et al., 2025) are used. When the rule databases are up-to-date, accurate, and properly configured, these tools can detect known wireless threats.

Signature systems work well on known types of attacks, but they are essentially reactive, i.e., they can only identify what they have seen before. Anomaly-based systems complement this by building a statistical baseline of normal network behavior and detecting any anomalies. In wireless scenarios where legitimate fluctuations, mobility of devices, and interference cause constant changes, however, such systems produce false-positive alerts that diminish the usefulness of operations (Ozkan-Okay et al., 2021). Neither includes machine-learning adaptive detection, automated anomaly scoring, or a human-readable explanation of threats, and does not provide the network administrator with information about the nature, severity, or recommended action for events that are detected.

3.2.2 Limitations of the Existing System

Although the wireless IDSs are widely deployed, the proposed system is motivated by the following 5 critical drawbacks in existing wireless IDSs:

- **A lack of ability to identify unknown or new attacks**: Signature-based IDS cannot detect variants of the attacks that aren't yet included in the rulesets, or new attacks that haven't been seen before. Wireless attacks like adaptive deauthentication (variation in frame timing, source address) can bypass even signature matching (Vinayakumar et al., 2019).
- **Frequent false positives**: When devices move, there are environmental distractions, and there is network congestion; these are all legitimate variations in a wireless signal that can be misinterpreted as being an intrusion event, which leads to alert fatigue for administrators (Acka et al., 2025).
- **Manual maintainability overhead**: Signature databases are dependent on regular manual updates to be effective against the constantly evolving threat, and there are time gaps in the effectiveness between the evolution of a new threat and when it is included in the signature database.
- **No adaptive learning**: Traditional IDS does not learn from previous traffic patterns and cannot get better at detecting the traffic over time. Detection decisions are taken independently using specific rules (Sowmya et al., 2023).
- **Lack of explainability and threat guidance:** Today's tools provide only technical output such as alerts, packet data, and/or rule matches; none explain what was detected, why it is a threat, or what action the administrator should take in response (Yellepeddi et al., 2024).

3.2.3 Analysis of the Proposed System

The suggested system is an AI-based wireless network intrusion detection system, which directly overcomes each of the five limitations mentioned above, and is web-based. The proposed system uses a machine learning-based detection architecture to learn patterns from the AWID3 wireless benchmark dataset, and then uses this learning to accurately classify network traffic records, without requiring manual maintenance of rules.

The system alleviates the lack of labelled anomalous traffic by using an unsupervised anomaly detection algorithm called Isolation Forest for identifying traffic records that structurally differ from normal traffic without needing labelled examples of anomalous traffic. It handles high false positive rates because of the ensemble effect of Random Forest: voting from multiple decision trees to create stable and reliable classification. It takes the manual maintenance out of the picture by using trained models that can be retrained when new data is available, instead of using static rule databases. It offers adaptive learning by means of model training that learns a generalized pattern from labelled traffic examples, which are then encoded into the decision boundaries of the classifier.

The explainability gap is solved by an integrated explanation engine, which is composed of two parts: SHAP analysis and a rule-based expert system. SHAP calculates the contribution of each traffic feature towards the classification decision for each detected intrusion, and the top features that contributed to the classification decision will be identified. The expert system then applies the feature contributions using attack-specific condition-action rules to produce a structured plain-English explanation of the attack type, how it works, the potential impact, and the recommended response actions. The system is connected to a Flask-based website, which can be opened in any browser and shows the results on a fake dashboard in near-real-time mode.

3.3 Dataset Description and Preprocessing

3.3.1 Dataset Overview

The main dataset used for training and evaluation was the AWID3 (Aegean WiFi Intrusion Dataset version 3). AWID3 is a benchmark dataset tailored for wireless intrusion detection and is a representation of real IEEE 802.11 Wi-Fi network traffic, both in normal and attack modes. The characteristics of 802.11 management frames are not captured by other datasets like NSL-KDD or CICIDS2017, which are general-purpose datasets, and it is this characteristic that makes it more suitable for this project than such datasets, which are not designed to capture the characteristics of wireless protocols (Ozkan-Okay et al., 2021).

The AWID3 dataset was used to select three types of attacks. There were 33 CSV files in the Deauth folder, 41 CSV files in the Disas folder, and 40 CSV files in the RogueAP folder. The dataset comprises 33 feature columns that contain information about the IEEE 802.11 frame-level attributes such as radiotap signal measurements, WLAN frame control fields, frame timing information, and radio channel characteristics. The categories of attacks in this project are summarized in Table 3.1

Table 3.1: AWID3 Dataset Attack Categories

Attack Category	Folder	CSV Files	Description
Deauthentication	Deauth/	33	Forged deauth frames forcing client disconnection from AP
Disassociation	Disas/	41	Forced disassociation from wireless access points
Rogue Access Point	RogueAP/	40	Fake AP impersonating legitimate access points (evil twin)
Normal	Normal samples	N/A	Legitimate 802.11 wireless traffic representing baseline behavior

3.3.2 Train-Test File Split

In order to ensure that the model is being tested on sessions that it was not trained on, the file-level 80/20 training/test set split was adopted. The files that were assigned later numbers in the attacks directory were labeled as tests, and the testing set was designed in such a way that it would have actual attacks present in order not to be empty. The specific file allocation used is indicated in Table 3.2.

Table 3.2: File-Level Train-Test Split

Folder	Total Files	Training Files	Test Files	Test File Names
Deauth/	33	31	2	Deauth_31, Deauth_32
Disas/	41	38	3	Disas_38, Disas_39, Disas_40
RogueAP/	40	37	3	RogueAP_37, RogueAP_38, RogueAP_39

3.3.3 Preprocessing Steps

Raw CSV files were read in with pandas and split into train and test DataFrames. The following preprocessing operations were applied in order to ensure data quality and suitability for machine learning:

- Missing Value Handling: Columns with data greater than 50% missing were dropped. The remaining missing values were filled with the column median values, which were calculated from the training set only and then used for both training and test sets.
- Infinite Value Replacement: Infinite values that appeared in the calculations of the number of packets were replaced with NaN and were then imputed with the same training column medians.
- Label Encoding: The target label column was encoded using LabelEncoder fitted on the training labels only; the values of the class (Normal, Deauth, Disas, RogueAP) were mapped to integer values for training the model.
- Attack classes were undersampled to balance the training set with the natural class distribution of the test set, which was ~92.6% Normal. This approach yielded more realistic decision boundaries than the artificial 50/50 balancing, enhancing generalization over real wireless traffic distributions (Ikeri, 2025).
- Feature Scaling: StandardScaler was fitted on the training features and applied to both training and test sets, with no information from the test set used to alter the scaling parameters.
- Train-Test Separation: The entire preprocessing pipeline was executed separately on the Training and Test DataFrames. There was no test data used to fit any preprocessing object.

3.3.4 Training Data Summary

After preprocessing and class distribution matching, the final training data set consisted of 205,730 vectors from 91 training files from each of the 3 attack folders and their associated normal traffic capture files. The test set consisted of 313,193 rows across 8 held-out files, maintaining the class distribution found in a real set of wireless network captures. These two partitions are summarized in Table 3.3.

Table 3.3: Training and Test Dataset Summary

Partition	Files	Total Rows	Normal Rows	Attack Rows
Training	91	205,730	~190,706 (92.6%)	~15,024 (7.4%)
Test (Held-Out)	8	313,193	290,150 (92.6%)	23,043 (7.4%)

3.4 System Requirements Specification (SRS)

3.4.1 Functional Requirements

The following functional requirements the proposed system should be able to meet:

- The system should support uploading a CSV file that has wireless network traffic records in the AWID3 format via a web-based interface by a network administrator.
- The system shall preprocess the data that is uploaded. It shall use missing value imputation in case of missing values, replace infinite values with the median for that column, normalize numerical features with StandardScaler, and encode the label categorical variables with LabelEncoder.

- The system shall deal with class imbalance in the training set, undersampling attack classes proportionately from the training set so that its class distribution is approximately 92.6% Normal.
- The system will train a 4-class Random Forest classification algorithm on the preprocessed AWID3 training data set.
- This shall train an Isolation Forest model using only Normal rows to detect anomalies unsupervised; structurally unusual rows will be highlighted as potential threats.
- All Random Forest predictions shall be given a confidence threshold of 80% with any record that has a maximum class probability below this threshold shall be assigned Normal instead of alert.
- The system should simulate a near real-time stream of traffic to update the dashboard with classification results, confidence scores, and anomaly flags as traffic is processed, in order of arrival.
- The system should use SHAP Tree Explainer to identify the most important features contributing to the intrusion, and feed these results into a rule-based expert system, which should be structured to produce a plain-English explanation of the intrusion, including the mechanism of the attack, its impact on the system, and the recommended response for the user.
- The system shall provide a real-time dashboard that will display total records processed, total alerts, attack type distribution charts, anomaly counts, model confidence scores, and all threat explanations.
- The system should evaluate and print the accuracy, precision, recall, and F1 score of the Random Forest model on the test set that was not used in the training.

3.4.2 Non-Functional Requirements

The following non-functional requirements the proposed system has:

- **Performance:** With the AWID3 test set, the classification accuracy of the Random Forest model will not be less than 95%. During a simulated real-time streaming mode, at least 100 traffic records shall be processed per second.
- **Usability:** The usability of the web interface shall be simplified and made understandable for a network administrator who may lack knowledge in machine learning. All detection information and threat descriptions shall be displayed in a clearly readable language and visual organization.
- **Scalability:** The system shall be able to process up to 500,000 traffic records per CSV file without failure, without noticeable performance degradation, using chunked processing and/or data sampling, when necessary.
- **Reliability:** The system should operate consistently and correctly during the entire period of an analysis session, and should be robust enough to cope with flawed or incomplete CSV rows without crashing.
- **Security:** No raw traffic data will be sent to external services. All model inferences, computation of explanations, and generation of explanations shall be performed in the confines of the local system environment.
- **Portability:** It shall be able to run on any machine that has Python 3.8 or greater installed, and the only proprietary software or specialised hardware required will be open-source packages distributed via pip.

3.5 System Modelling Using UML

To ensure consistency in how the system is structured and behaves, Unified Modeling Language (UML) diagrams are used. The three types of diagrams are a use case diagram (to display the interactions between the users and the system), an activity diagram (to show the processing flow), and a class diagram (to show the structure of the system as a set of objects, their attributes, and their relationships).

3.5.1 Use Case Diagram

The use case diagram shows the main actors associated with the use cases, the Network Administrator using the Flask Web Console to perform the operations. The six use cases are: Upload Traffic File - Submitting a CSV file of wireless network traffic for analysis; View Real-Time Dashboard - Displaying real-time analysis results as they stream; View Detection Results - Displaying full classification report, attack type distribution, and confidence scores; View Threat Explanation - Showing plain-English descriptions for each detected intrusion by the expert system and SHAP; Model Performance Metrics - Viewing the accuracy, precision, recall, and F1-score for the random forest classifier; and Session History - Accessing detection session records from the SQLite database. Figure 3.1 shows the use case diagram.

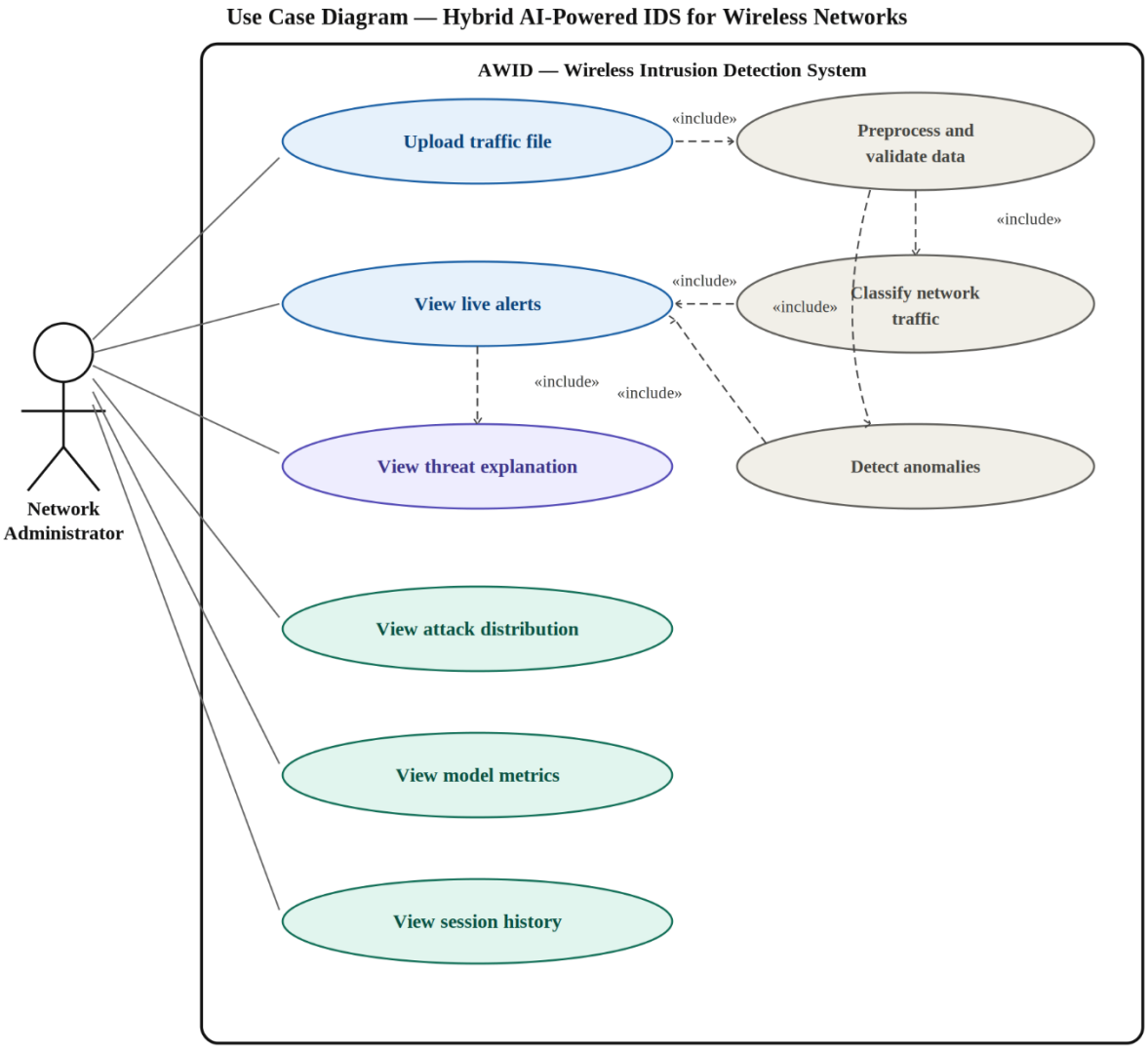

Figure 3.1: Use Case Diagram of the Proposed AI-Powered IDS

3.5.2 Activity Diagram

The Activity diagram represents the entire sequence of events in the process of the system, starting from user interaction to output. The flow starts by opening the Flask application (built by the network administrator) and uploading a CSV file with wireless traffic records in the AWID3 format. The system checks the format of the file and starts the pipeline for processing the data, which includes imputation of missing data, replacement of infinite values, numerical normalization with the fitted StandardScaler, and encoding of labels with the fitted LabelEncoder. Passed to the detection engine, which loads trained model files from disk.

Each record is inserted one at a time into the detection phase. Each record is labeled with a class label and a confidence score by the Random Forest classifier, and a separate anomaly score is calculated by the Isolation Forest. When the Random Forest confidence is greater than 80% and an attack category is mentioned by the label, the explanation engine is activated: SHAP TreeExplainer calculates feature contributions for this prediction, and the expert system applies the respective rules for each attack type to create a structured explanation in plain English. The dashboard automatically refreshes with the results of each record, and when finished, the system displays performance metrics and stores session results in the SQLite database.

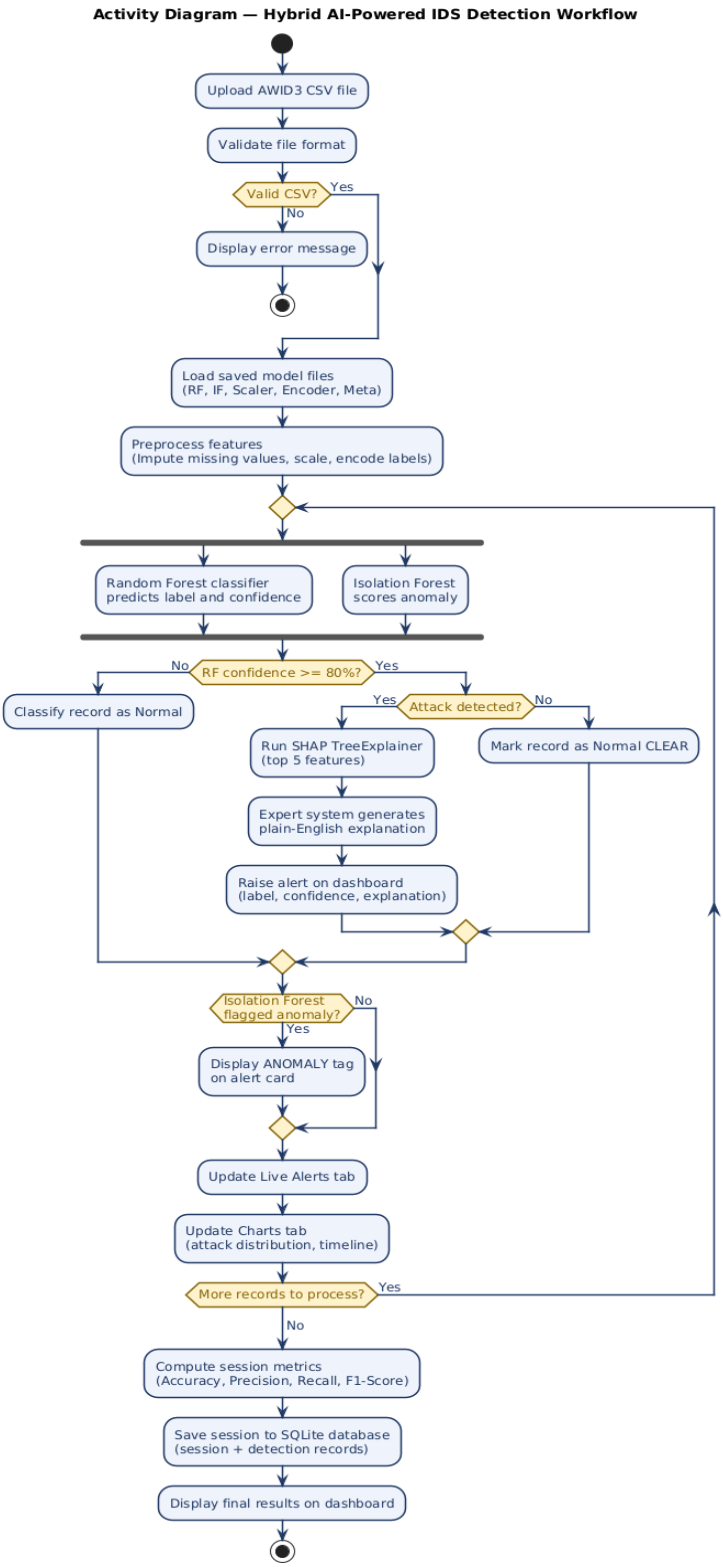

Figure 3.2: Activity Diagram of the System Detection Workflow

3.5.3 Class Diagram

The class diagram illustrates the basic classes of the system and how they are related. The DataLoader class is used to read and validate the uploaded CSV file and returns structured pandas DataFrames. All missing values, infinite value replacement, feature normalization with StandardScaler, and label encoding with LabelEncoder are done by the Preprocessor class, which was fitted only on the training data. The DetectionEngine class is an instance of a RandomForestClassifier and an IsolationForest, and provides methods to train, predict, score anomalies, and to evaluate the performance of the detection engine. The ExplanationEngine class is a wrapper around the class SHAP TreeExplainer and the rule-based expert system, which takes a detected attack label, confidence score, and feature vector, and provides a structured plain-English explanation. The DashboardController class is the class that will coordinate all other classes, handling the routes of Flask and orchestrating the end-to-end pipeline. The SessionStore class is responsible for handling the operations of the SQLite database that stores and retrieves detection session records.

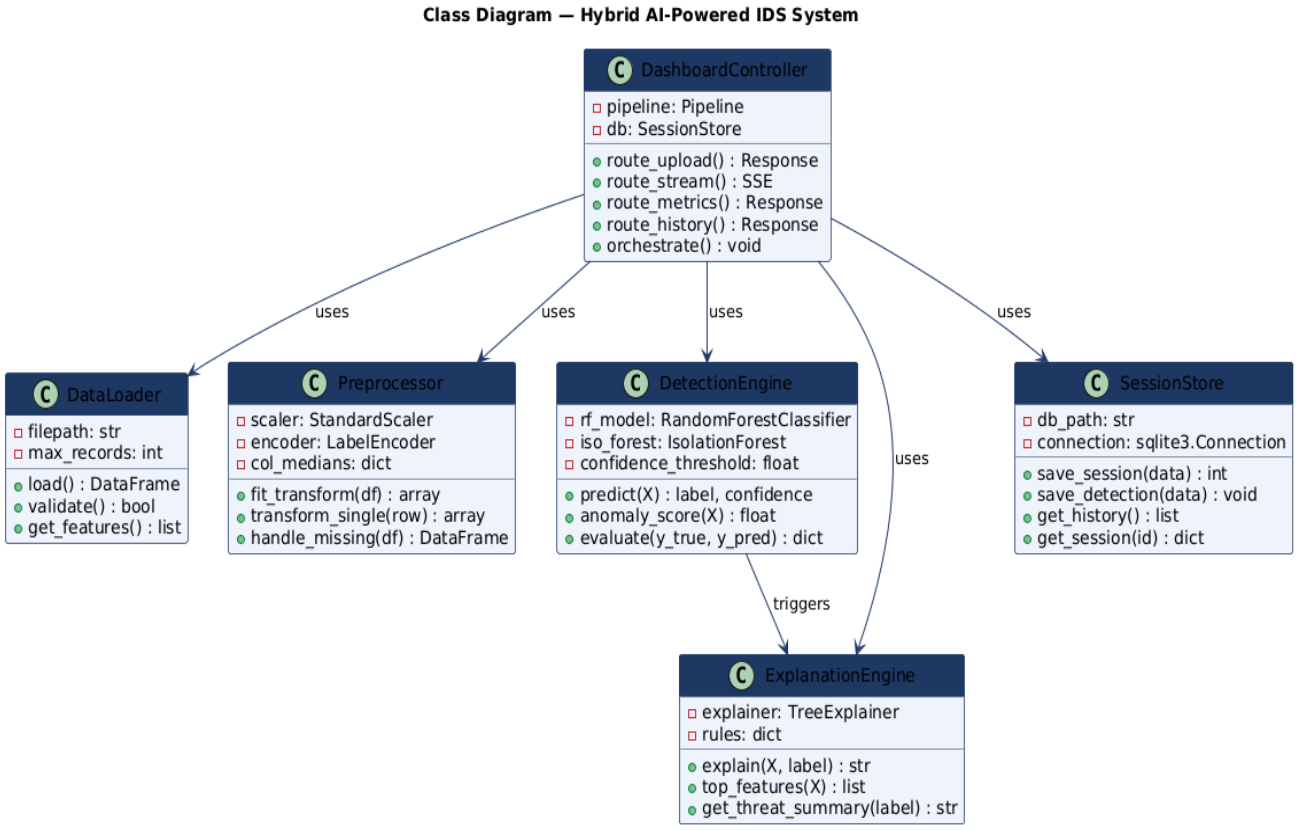

Figure 3.3: Class Diagram of the Proposed System

3.6 System Design

3.6.1 Input Design

The most critical data to the system is the CSV file containing the wireless network traffic data in the format of the AWID3 dataset structure. Each row corresponds to a single traffic record in a network, and each column is a feature of the 802.11 frame, such as frame type or subtype, duration, source address category, sequence number, signal strength, and protocol-specific attributes derived from raw 802.11 captures. The final column will be Normal, Deauth, Disas or RogueAP for the class label.

The user enters the input layer via a web form on the Flask dashboard. The form has a well-labelled button for upload that only takes CSV files. A parameter field is provided, and the number of records to be processed can be specified; a suggested default value is 1,000 for interactive sessions. When it's sent, it's read by the Flask backend and given to a DataLoader class to validate and load it.

3.6.2 Output Design

System output comes in four types, all of which are presented in the Flask web dashboard. The first is the real-time alert feed, which is a scrolling list with the alerting record index, class label, Random Forest confidence score, and Isolation Forest anomaly flag. Records that are "normal" are shown with a 'green clear' check mark; records that are "in attack" are shown with a 'red alarm' check mark. Any data marked by the Isolation Forest as unusual will have an extra ANOMALY tag. The second output is the attack distribution chart [bar chart] that displays the number of each attack type detected, as a percentage of normal traffic, and is continuously updated as the processing continues. An accompanying donut chart shows the site's overall ratio of normal to malicious traffic. The third is the model performance panel, which is a summary table showing the

accuracy, precision, recall, and F1-score of the Random Forest model, plus the summary statistics of the Isolation Forest. The fourth output is the threat explanation panel, in which for every intrusion detected, a structured card is displayed that includes the label of the type of attack, the timestamp of the detection, a confidence score, whether or not there is an anomaly, and an explanation rendered by the expert system and SHAP engine on the attack mechanism, potential impact, and recommended response in plain English.

3.6.3 Database Design

It relies on a lightweight SQLite database to store the records of the detection sessions so that the administrator can check the results of an analysis without reprocessing the files. SQLite is selected because it supports deployment without any configuration, is file-based, and has built-in support from the Python sqlite3 library, which makes it a good choice for a one-user desktop application. The system does not use a login mechanism, so there are no tables for user authentication in the database.

There are two tables in the database. The Session table contains one record for each analysis session and fields include: session_id (INTEGER, PRIMARY KEY, AUTOINCREMENT), session_date (TEXT — ISO format timestamp), filename (TEXT), total_records (INTEGER), and total_alerts (INTEGER). The Detections table contains one record per detected intrusion with the following fields: detection_id (INTEGER, PRIMARY KEY, AUTOINCREMENT), session_id (INTEGER, FOREIGN KEY on Sessions), record_index (INTEGER), attack_type (TEXT — one of Deauth, Disas, RogueAP), confidence_score (REAL), anomaly_flag (INTEGER — 0 or 1), explanation (TEXT — the expert system and SHAP generated narration), and top_features (TEXT — JSON-serialised list of top SHAP feature contributions). To keep the size of the database down, normal traffic records are not stored. One Session can have many Detections.

3.6.4 System Architecture Design

This system has been designed using a three-tier web application architecture, consisting of the presentation tier, the processing tier, and the data tier. This architectural approach is quite well known and helps keep all tiers independent of each other.

The presentation tier is made up of the Flask HTML templates and JavaScript that are used to display the web dashboard in the administrator's browser, as well as handling file uploads and updating the dashboard in real-time, and rendering charts using Chart.js. The processing tier consists of Flask backend application running on local Python server, which consist of five modules: DataLoader for ingesting data from files and validating; Preprocessor for processing all data cleaning and transformation; DetectionEngine that includes the trained Random Forest Classifier and the Isolation Forest anomaly detector; ExplanationEngine that includes SHAP computation and application of rules from the expert system; and DashboardController that coordinates all Flask routes. No API calls – everything is processed locally. Data tier: the SQLite database file, the AWID3 CSV dataset files, and five saved model files (random_forest_model.pkl, isolation_forest_model.pkl, scaler.pkl, label_encoder.pkl, feature_meta.pkl), all serialised using Python's joblib library. Figure 3.5 shows the system architecture.

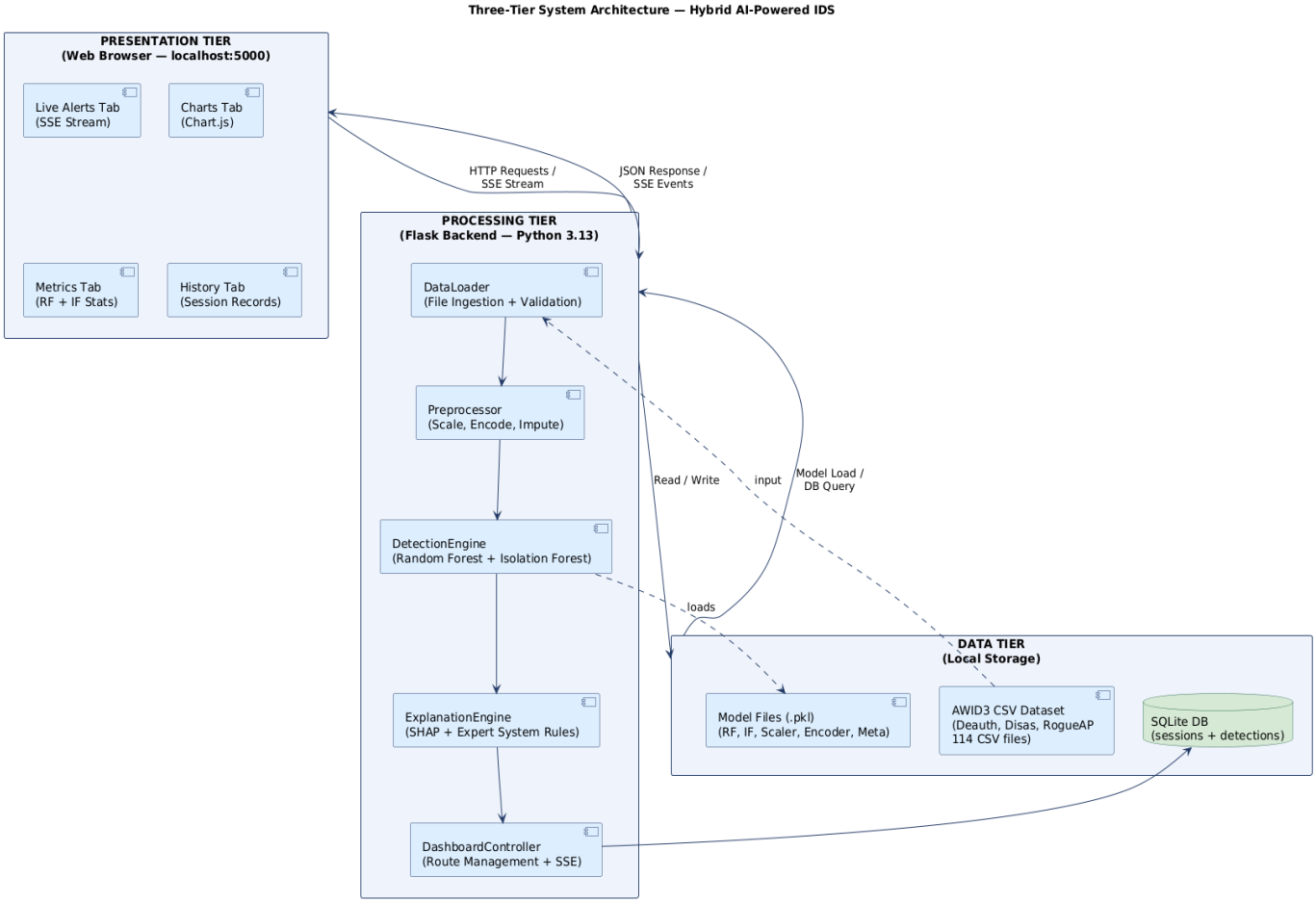

Figure 3.4: Three-Tier System Architecture Diagram

Table 3.4: Summary of System Components and Responsibilities

Component	Technology	Responsibility	Tier
Web Interface	Flask, HTML5, CSS3, Chart.js	File upload, dashboard display, and chart rendering	Presentation
DataLoader	Python, pandas	File validation and CSV ingestion	Processing
Preprocessor	pandas, scikit-learn	Missing values, normalisation, encoding, undersampling	Processing
DetectionEngine (RF)	scikit-learn RandomForestClassifier	Supervised multi-class attack classification	Processing
DetectionEngine (IF)	scikit-learn IsolationForest	Unsupervised anomaly detection	Processing
ExplanationEngine	SHAP TreeExplainer, Rule-based Expert System	Feature explanation and threat narrative generation	Processing
DashboardController	Flask routes, Python	Pipeline orchestration and route management	Processing
SessionStore	SQLite, Python sqlite3	Session and detection record persistence	Data
Model Storage	joblib (.pkl files)	Saving and loading trained model files	Data
AWID3 Dataset	CSV files (Deauth, Disas, RogueAP folders)	Training, validation, and evaluation data	Data

CHAPTER FOUR
SYSTEM IMPLEMENTATION

4.0 Introduction

The Hybrid AI-Powered Intrusion Detection System (IDS) for wireless networks implementation is detailed in this chapter. It explains the development environment, tools, libraries, and processes for the creation of the system, covering the preparation of the datasets, the creation of the models, the dual-model detection system, the expert system and SHAP explanation system, the Flask web application, and the integrated dashboard. The implementation is based on the system design specifications outlined in Chapter Three and is a fully working system that meets the proposed architecture.

The design has been realized with the use of Python version 3.13 programming language together with the Flask web application framework. The algorithm used is made up of two main classifiers, which are the supervised Random Forest model for attack classification and the unsupervised Isolation Forest model for anomaly detection. The literature review provided strong support for the proposed method due to high detection accuracy with the use of ensemble techniques such as the Random Forest approach (Saranya et al., 2020; Ikeri, 2025).. The explainability layer was achieved using SHAP (SHapley Additive exPlanations) and incorporating a rule-based expert system, which is one of the most constant issues faced in AI-driven intrusion detection studies (Acka et al., 2025; Amachaghi, 2025).

4.1 Development Environment and Tools

The system has been built on a Windows workstation. All implementation, testing, and evaluation have been done in-house, without depending on cloud services or any external infrastructure. A summary of tools and libraries used in every part of the system is available in Table 4.1.

Table 4.1: Development Environment and Tools

Component	Tool / Library	Purpose
Programming Language	Python 3.13	The Core implementation language
Web Framework	Flask	Lightweight web application server and routing
Machine Learning	scikit-learn	Random Forest, Isolation Forest, preprocessing pipelines
Explainability	SHAP (TreeExplainer)	Feature-level explanation of model predictions
Data Processing	pandas, NumPy	Dataset loading, feature engineering, class balancing
Visualization	Chart.js	Attack distribution charts and detection timeline
Database	SQLite	Embedded database for sessions and detections tables
Model Persistence	Joblib	Saving and loading trained model files (.pkl)
Frontend	HTML5, CSS3, JavaScript	Responsive dark-themed dashboard interface
Development IDE	Visual Studio Code	Primary development environment

4.2 Hybrid Detection Engine

4.2.1 Dual-Model Architecture

The heart of the system is a hybrid detection engine, which executes two parallel machine learning models on each traffic record, providing a complementary approach to the problem. The hybrid approach was selected because a known weakness of IDS research is that single-model systems are effective when the attacks they are designed to detect are known, but are ineffective when the attacks are not contained in the training distribution, meaning they are unfamiliar or uncommon. The hybrid learning approaches that take advantage of both supervised and unsupervised learning are the most powerful IDS design strategy, as shown by Ikeri (2025), and overcome the drawbacks of supervised and unsupervised approaches.

The dual-model architecture works as follows: A Random Forest (RF) classifier is used to classify known attacks by assigning each traffic record to one of four pre-defined labels. Isolation Forest is a model that looks at each record and assigns an anomaly score of its own for anomalous behaviour. All the predictions generated by the Random Forest are filtered by a confidence threshold of 80%, where the maximum probability of the class is less than that threshold, which means that the false positive rate in operational applications is lower than that of the other classifiers.

4.2.2 Random Forest Classifier (Supervised Model)

Random Forest model was selected as the key supervised classification algorithm. RandomForest is an ensemble approach, where several decision trees are created during training, and then the mode class of most of those trees is returned; there are three benefits to using the Random Forest approach: it will not be subject to overfitting, will perform well with class imbalance, and does not need much hyperparameter tuning (Saranya et al., 2020; Sowmya et al., 2023).

The model is trained with 100 decision trees (n_estimators=100, random_state=42) using the balanced training partition. The trained model was then saved to the models/ folder as random_forest_model.pkl using the joblib library. During the inference phase, the model outputs a class label and a probability distribution for each class. The highest probability is shown as the detection confidence score on the dashboard, along with each alert.

4.2.3 Isolation Forest (Unsupervised Anomaly Detector)

As the unsupervised part of the hybrid detection engine, the Isolation Forest algorithm was used. Isolation Forest works by randomly splitting the feature space and determining the time it takes to isolate a certain data point from the others. The fewer the number of partitions needed to isolate an anomalous instance, the lower score it will get for the anomaly. This method has low computational complexity, is scalable to large datasets, does not need labeled training data, and is resistant to class imbalance (Acka et al., 2025).

Isolation Forest was trained using only the Normal-labeled rows in the training partition, allowing it to learn a true baseline of normal wireless behavior. In detection, if the record is classified as an anomaly by the Isolation Forest, it is assigned an ANOMALY indicator on the dashboard, regardless of the outcome of the classification by the Random Forest, so that there is a second layer of detection which is effective against zero-day or previously unseen attack patterns (Vinayakumar et al., 2019).

4.3 Expert System and SHAP Explanation Engine

4.3.1 Motivation for Explainability

One of the major hurdles in implementing AI-powered IDS is understanding the rationale of the model's decisions. Many of the high-performing models are black-box models that just classify traffic, without giving any idea as to why a given traffic event was detected as malicious. This incomprehensibility impacts the analyst's trust and affects the incident response (Yellepeddi et al., 2024; Acka et al., 2025). This is where the explanation engine of this system comes in useful: it consists of two layers, namely SHAP feature analysis and a rule-based expert system, all realized as a module called 'explanation_engine.py'.

4.3.2 SHAP Integration

SHAP was integrated with a TreeExplainer interface optimized for ensemble tree models like Random Forest. SHAP distributes the value of the model's prediction among all features for each intrusion, using Shapley values computed from cooperative game theory. The explanation engine identifies the most important features for each prediction and presents them in natural language. The consistent top features on the AWID3 dataset were frame-level timing features, management frame subtype indicators, and signal strength measurements, offering a detailed, actionable features-level understanding of each detection decision for security analysts (Acka et al., 2025).

4.3.3 Rule-Based Expert System

A rule-based expert system was added on top of the SHAP outputs to convert the numerical feature contributions into textual threat narratives. The expert system then uses the condition-action rules, which are pre-defined and associated with each attack category, to generate a structured plain-text description that includes an attack mechanism, potential impact on the wireless network, recommended response actions for the wireless network administrator, and the Isolation Forest

anomaly assessment. It is a two-layered design that incorporates quantitative transparency via SHAP and qualitative operational guidance by expert system rules, directly addressing the interpretability gap identified by Acka et al. (2025) and Amachaghi (2025).

4.4 Evaluation

The trained hybrid model was validated on eight completely held-out test files, which were not used during any step of the training and pre-processing procedure. Evaluation was done at 80% confidence, which meant that predictions with a confidence less than 80% of the Random Forest confidence were considered Normal and not an Alert. The threshold-filtered assessment is based on the actual performance of this model when installed in the dashboard. Four standard metrics used in evaluation research of IDS, namely accuracy, precision (weighted), recall (weighted), and F1-score (weighted), were used to measure the performance (Saranya et al., 2020; Vinayakumar et al., 2019; Ikeri, 2025).

4.4.1 Overall Evaluation Results

Table 4.2 shows the results of the overall evaluation using all 8 test files (313,193 records). The accuracy obtained in this study is also in line with the reported results in the literature for the AWID dataset, as reported by Abdulhammed et al., as cited in Ozkan-Okay et al. (2021), who used Random Forest and obtained 99.64% accuracy.

Table 4.2: Overall Model Evaluation Results on Held-Out Test Set

Metric	Value	Description
Accuracy	99.44%	Proportion of all 313,193 test records correctly classified
Precision (weighted)	99.55%	Weighted average precision across all four classes

Metric	Value	Description
Recall (weighted)	99.44%	Weighted average recall across all four classes
F1-Score (weighted)	99.47%	Weighted harmonic mean of precision and recall
Confidence Threshold	80%	Minimum RF probability required to raise an alert
Test Records	313,193	Total records evaluated across 8 held-out files

4.4.2 Evaluation Metrics

The performance of the Hybrid Random Forest classification algorithm was also analyzed using three popular visualization methods including: the confusion matrix, ROC (Receiver Operating Characteristic) curve, and Precision-Recall curve. The visualization method gives us an understanding of the behavior of the classification algorithm for all the four classes – Normal, Deauth, Disas and RogueAP, in more detail than what could be understood by just numbers. Figures 4.1,4.2,4.3 show the Precision-Recall curve, Confusion Matrix, and ROC curves, respectively.

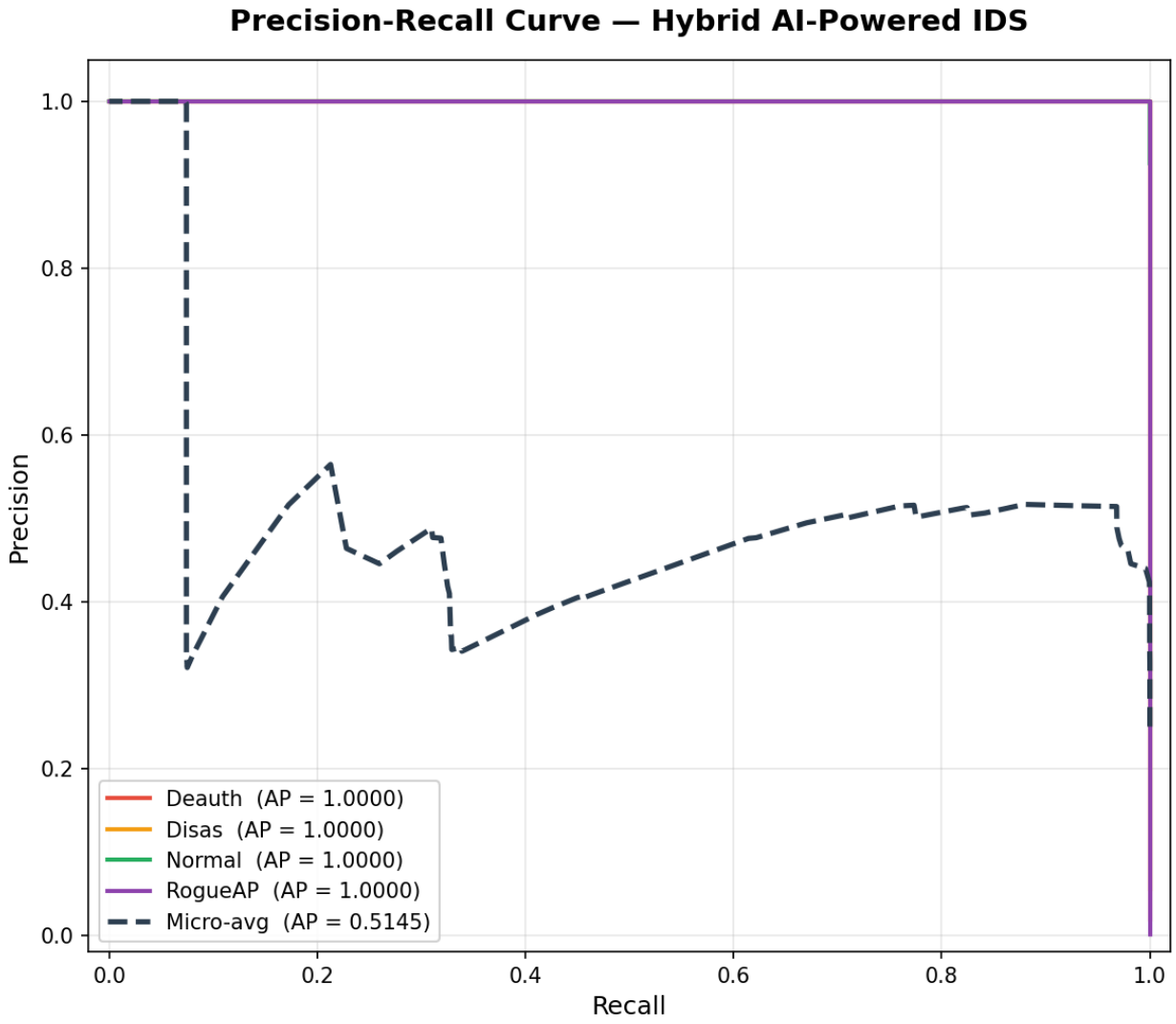

Figure 4.1: Precision-Recall Curve

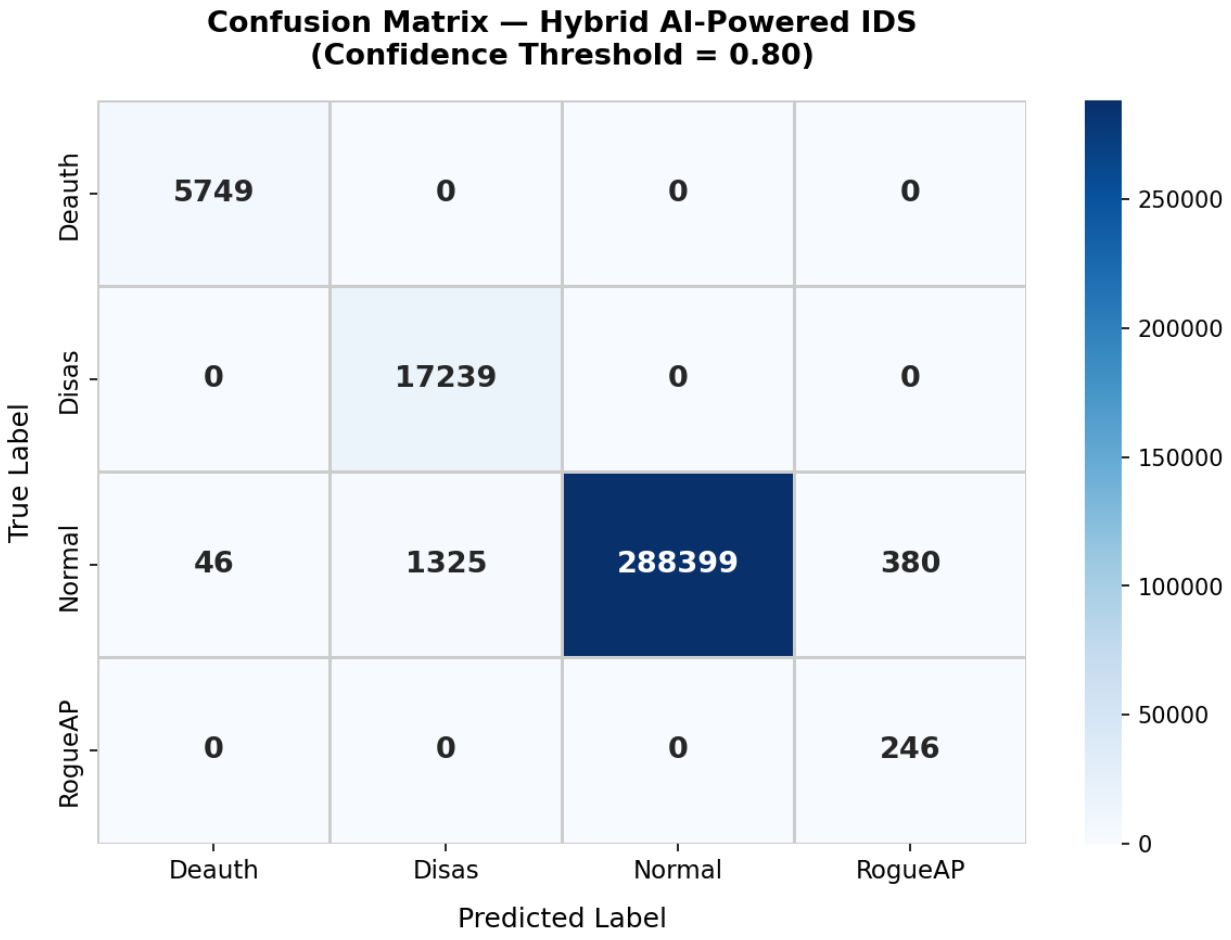

Figure 4.2: Confusion Matrix

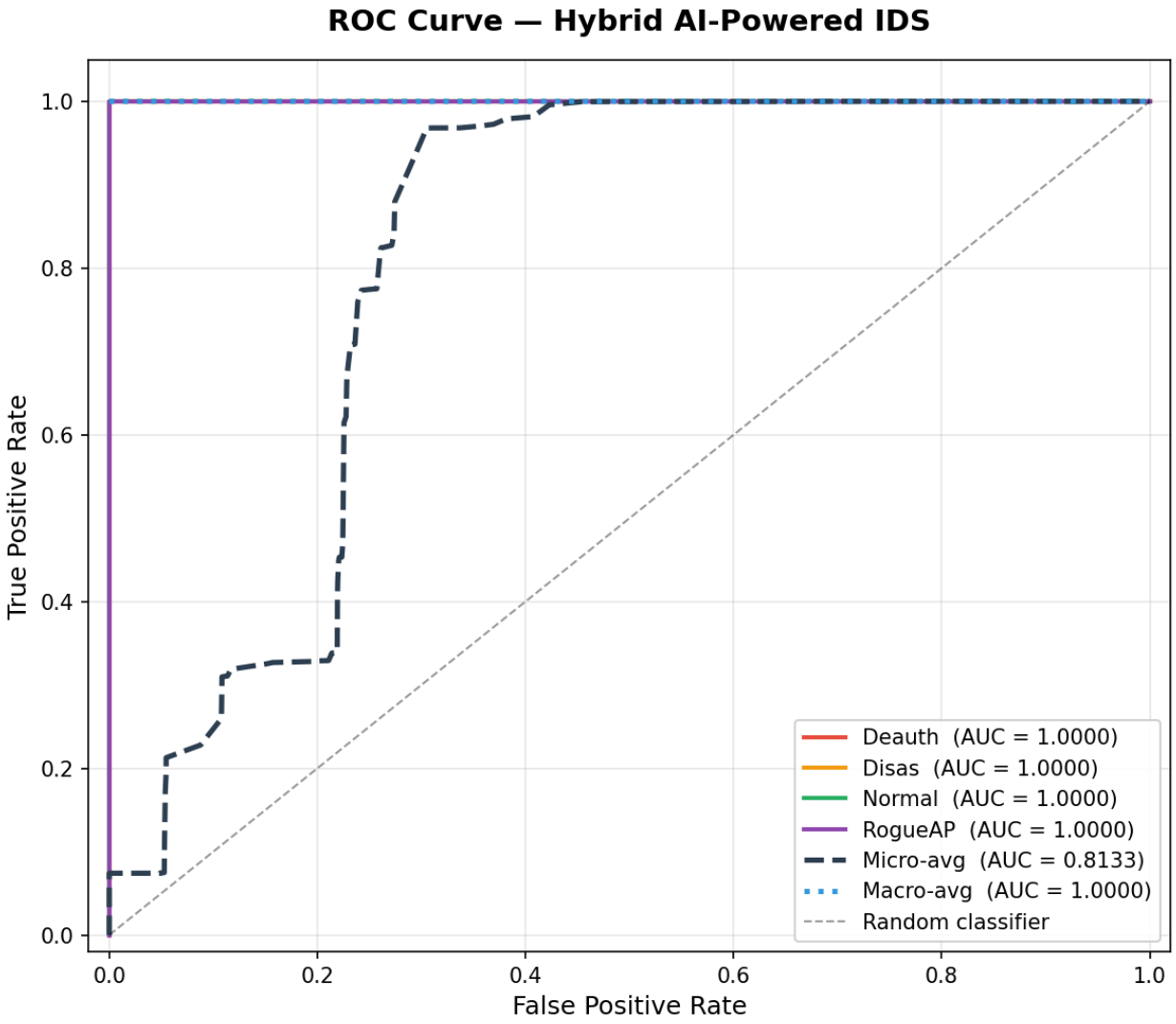

Figure 4.3: ROC Curve

4.4.3 Per-File Dashboard Results

Table 4.3 represents a set of live dashboard metrics collected as each of the eight test files was uploaded and passed through the system alone. These results are the real-time operational results that are displayed by the dashboard interface and communicated to the network administrator. The values reported in the metrics are not the same across files and demonstrate that the system does not return static metrics, but rather metrics specific to each session.

Table 4.3: Per-File Dashboard Evaluation Results

Test File	Attack Type	Records	Alerts	Accuracy	Precision	Recall	F1-Score
Deauth_31.csv	Deauth	1,000	114	98.20%	99.00%	99.10%	99.24%
Deauth_32.csv	Deauth	1,000	43	99.60%	99.63%	99.70%	99.70%
Disas_38.csv	Disas	1,000	198	98.70%	99.46%	99.54%	99.60%
Disas_39.csv	Disas	1,000	158	96.50%	97.28%	96.50%	96.68%
Disas_40.csv	Disas	1,000	71	99.70%	99.71%	99.70%	99.70%
RogueAP_37.csv	RogueAP	1,000	5	99.70%	99.88%	99.70%	99.76%
RogueAP_38.csv	RogueAP	1,000	5	99.60%	99.92%	99.60%	99.73%
RogueAP_39.csv	RogueAP	1,000	1	99.90%	100.00%	99.90%	99.95%

The system does not score uniformly on all inputs, with Disas_39.csv giving the lowest accuracy of 96.50% among all the test files. These were found to be significantly different across files, ranging from 1.4% for Disas_38.csv to 54.9% for RogueAP_39.csv, as the unsupervised anomaly detection component operates independently of one another, reacting to structural changes in the traffic pattern of each capture session.

4.5 System Integration and Testing

All 5 model files: random_forest_model.pkl, isolation_forest_model.pkl, scaler.pkl, label_encoder.pkl, and feature_meta.pkl were generated after training the model, and the explanation engine and the Flask application were then combined into a single end-to-end pipeline. The integration tests confirmed that Flask routes invoked the hybrid detection engine, and returned streaming Server-Sent Events to the dashboard; SHAP explanations were generated without error for each of the four prediction classes; detection results were persisted to the SQLite database with all required fields; the dashboard showed live alerts, charts, metrics, and history for all eight test files; and all performance metrics were computed and rendered correctly per uploaded file.

The application was successfully validated to run without errors using the Python development server at “localhost:5000”. All the main dashboard features (i.e., the detection alert cards with explanation cards from expert systems, the metric indicator panels, the anomaly flag indicators, the attack distribution bar charts, the normal vs malicious donut charts, and the cumulative detection timeline) were checked for validity both across all three attack categories and across all eight test files.

4.5.1 Dashboard User Interface

The web dashboard has been developed with an HTML5/CSS3/JavaScript responsive interface using a dark-themed color scheme to minimize eye strain over long periods of time when using the dashboard during security operations. Detection information is displayed in four distinct and easy-to-identify navigation tabs: Live Alerts, Charts, Metrics, and History.

The Live Alerts tab provides a scrolling list of detection events in almost real-time as they are received sequentially. Each alert card contains the expert system and SHAP-generated plain-English threat explanation, along with the prediction of the attack label, detection confidence

score, and the anomaly flag status. Normal records are highlighted with a green CLEAR indicator, and the attack records are highlighted with red warning badges. Records marked by the Isolation Forest have an extra ANOMALY tag that is not dependent on the Random Forest classification.

The Charts tab will generate an attack type distribution bar chart, which indicates the number of attacks of each detected attack type, as well as a donut chart that indicates the ratio of normal and malicious traffic, and a timeline chart that shows the cumulative number of attacks detected over time. The Metrics tab shows four important metrics (accuracy, precision, recall, and F1-score) calculated from the current session's predictions and actual labels in the uploaded file, as well as statistics about the anomalies detected by the Isolation Forest algorithm: total number of anomalies detected and overall anomaly rate. The History tab allows you to access all previous detection sessions in the SQLite database on a page-by-page basis. Figure 4.4 shows the system dashboard.

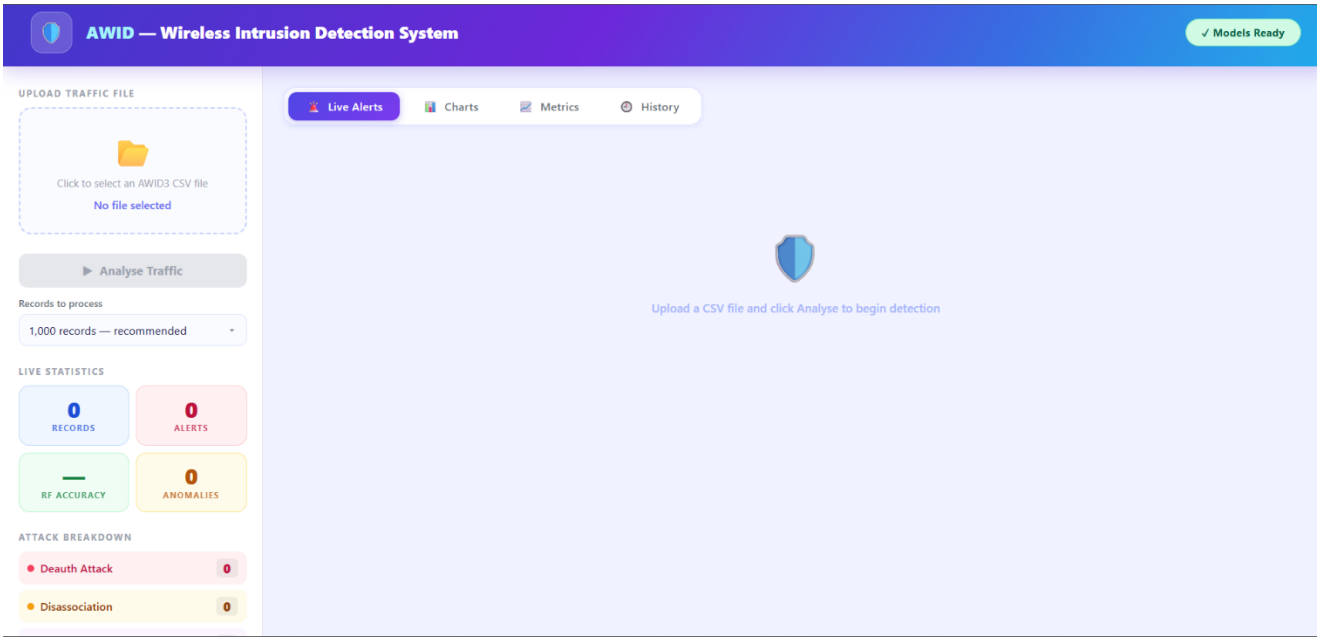

Figure 4.4: System Dashboard

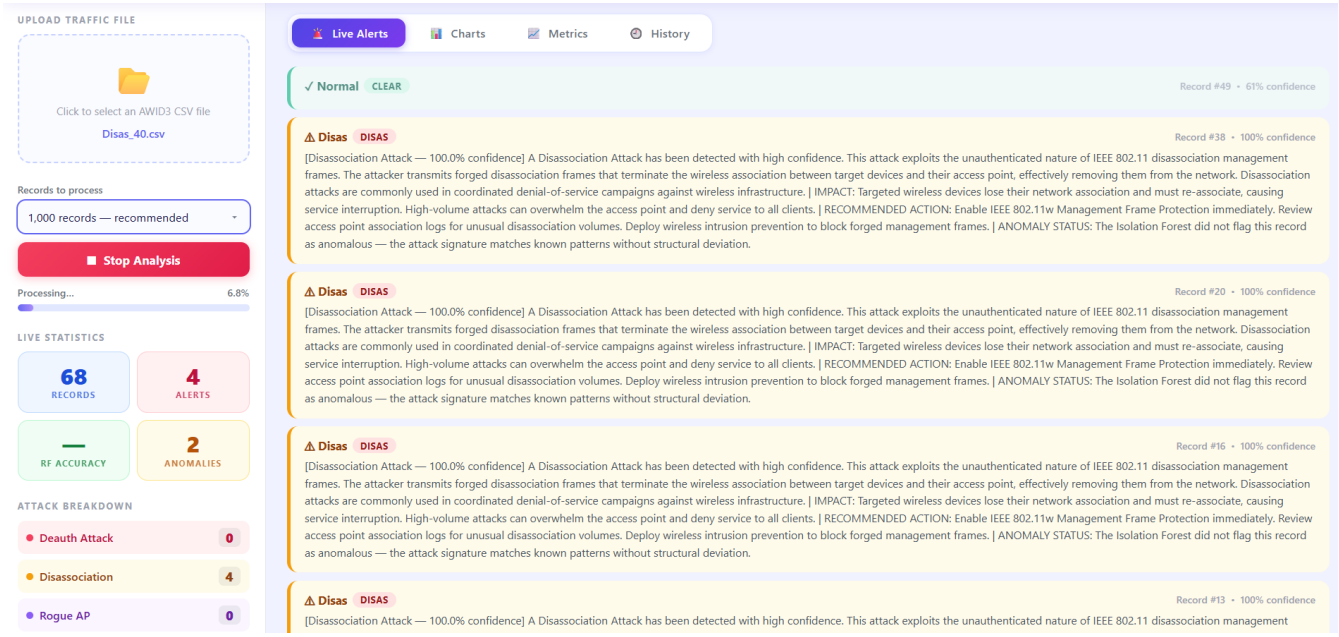

Figure 4.5: Explanation Interface

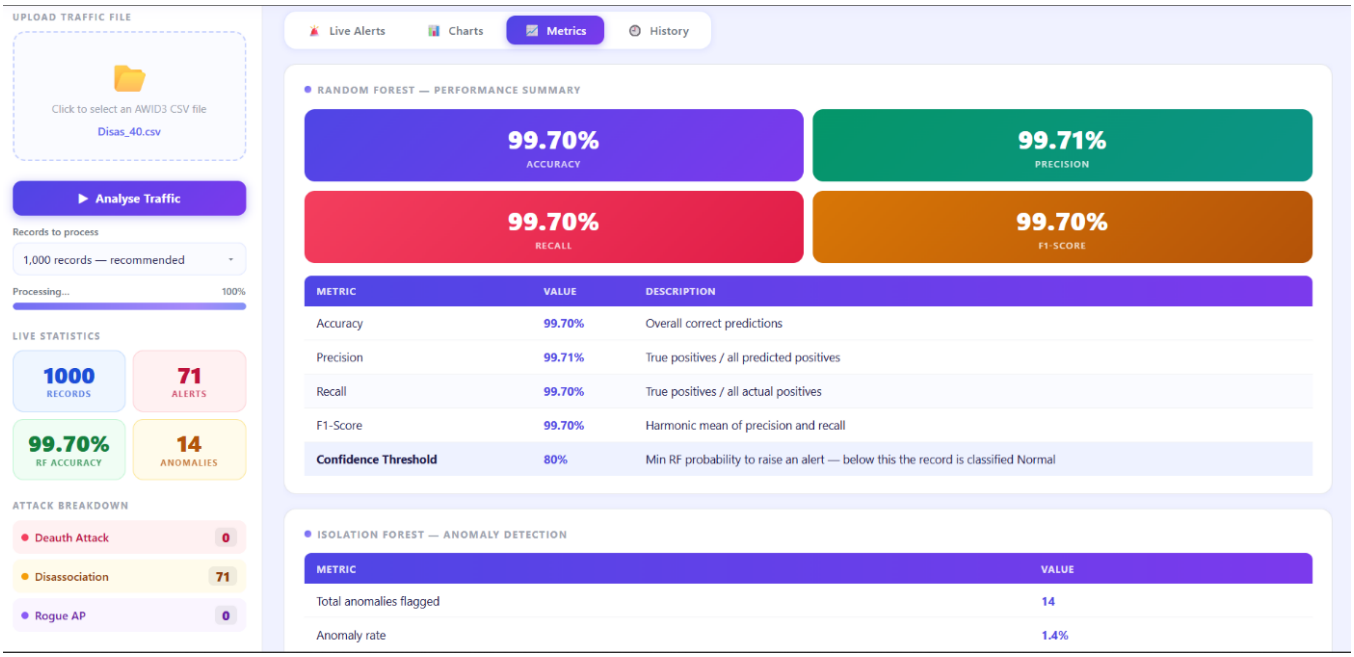

Figure 4.6: Evaluation Metrics

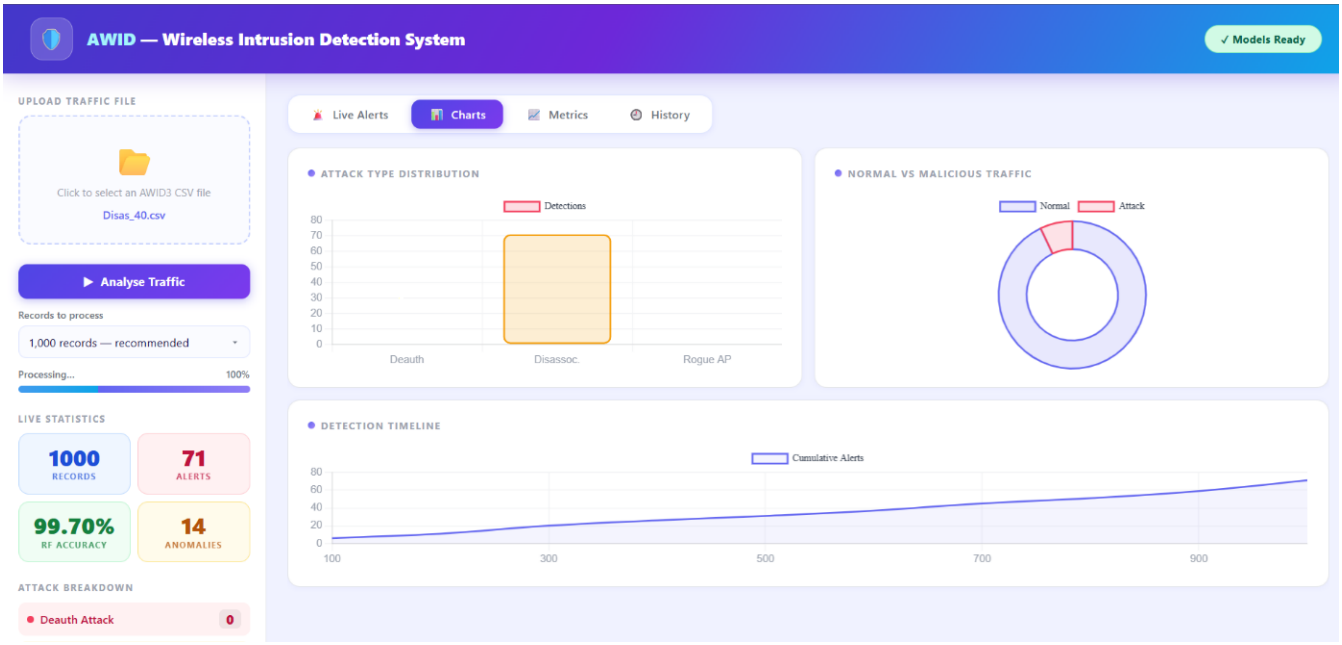

Figure 4.7: Detection Chart

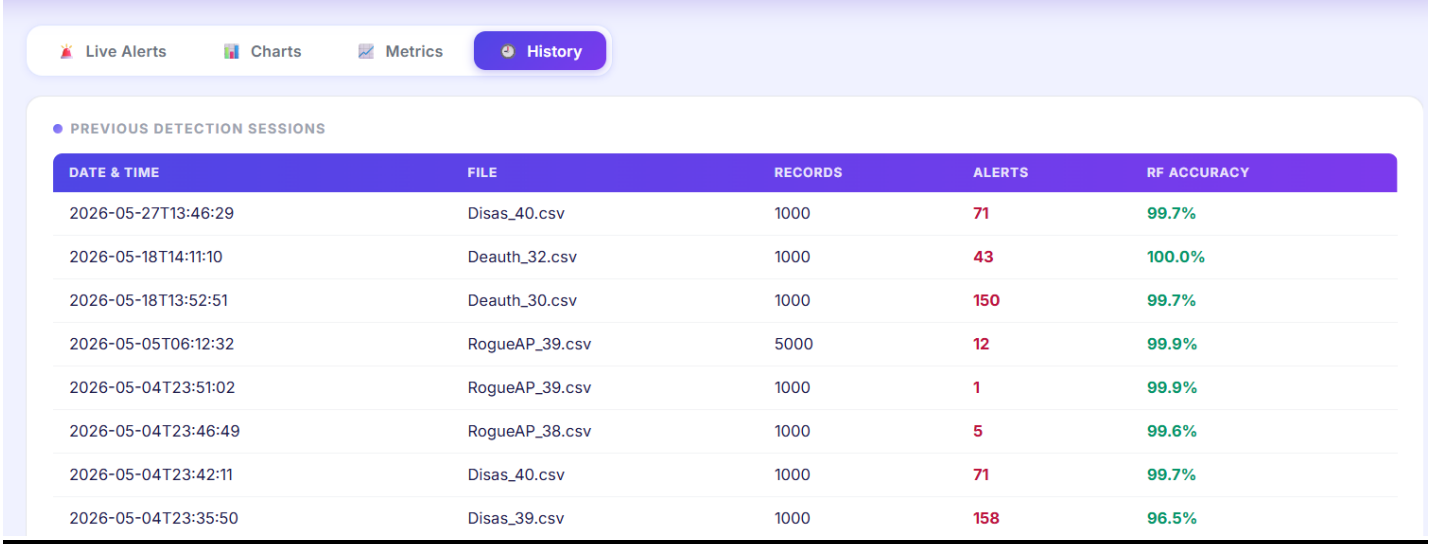

Figure 4.8: Session History

4.52 Dashboard Evaluation

The primary dashboard appears when the app is first opened (as seen in Figure 4.4). The left panel displays the upload interface, record selection dropdown, live statistics over records processed, records raised, RF accuracy, the number of anomalies, as well as an attack breakdown panel that displays the number of detections per attack type. Before starting an analysis, the Models Ready indicator verifies that the model files saved are loaded.

The Live Alerts tab for Disas_40.csv is displayed in Figure 4.5 when this file is analyzed. Each alert card includes the predicted attack label, confidence score, record number, and a structured plain-English explanation provided by the SHAP engine and expert system, which contains the attack mechanism, network impact, recommended response actions, and Isolation Forest anomaly status. Normal records are marked with a green CLEAR badge, and Attack records are marked with colour-coded attack labels to help distinguish benign traffic from malicious traffic at a glance.

The Metrics tab in Figure 4.6 displays the metrics after processing 1,000 records from Disas_40.csv. With the 80% threshold applied to the Random Forest classifier, the accuracy rate was found to be 99.70%, the precision rate was 99.71%, the recall rate was 99.70%, and the F1 score was calculated to be 99.70%. The 14 anomalous records were detected by the Isolation Forest, which corresponds to an anomaly rate of 1.4%.

The Charts tab of the same session appears in Figure 4.7. The Attack Type Distribution bar chart indicates 71 Disassociation detections, 0 Deauth, and 0 Rogue AP detections. The Normal vs Malicious Traffic donut chart shows the proportion of threats in the session, while the Detection Timeline presents a linear escalation of the number of alerts across the 1,000 records analyzed, indicating a consistent frequency of attacks throughout the file.

The History tab will list all the detection sessions that occurred in the past, stored in the SQLite database, as shown in Figure 4.8, each with the date, file name, records processed, alerts raised, and RF accuracy.

CHAPTER FIVE

SUMMARY, CONCLUSION, AND RECOMMENDATIONS

5.1 Introduction

The research project draws conclusions based on the results presented in Chapter Four, and makes recommendations for further work. The chapter revisits the aim and objectives mentioned in Chapter One and assesses how well the implemented system accomplished the objectives, and what the results mean for the wider area of wireless network intrusion detection. It also shows where the current implementation is limited and suggests some options for the future extension and improvements of the system.

5.2 Summary of the Study

The background and motivation for this research were laid out, and the critical security vulnerabilities that exist in IEEE 802.11 wireless networks were identified. Wireless networks are vulnerable to several protocol-level attacks due to their open broadcast nature, such as deauthentication and disassociation attacks, and rogue access point impersonation, which are not handled well by traditional IDSs. The drawbacks of other tools like Snort, Suricata, and Kismet, with respect to detecting novel threats, high false positive rate, and giving meaningful threat guidance, were identified as the main problems motivating this study. The goal was to design and develop a hybrid wireless network intrusion detection system with a dual machine learning approach and an integrated explainability engine.

The literature review was conducted in five theoretical areas that were of relevance to the proposed system, as reviewed in Chapter Two. The review has identified that, overall, there is no better detection method than Random Forest when tested on a variety of IDS benchmark datasets, that a

hybrid approach is the most comprehensive detection method, and that explainability mechanisms for existing AI-based IDS are a significant challenge for operational deployment. AWID3 was chosen as the best benchmark for the purpose of this project, as it possesses genuine IEEE 802.11 wireless traffic, along with the fact that it only includes the traffic relevant to the three attacks. This project fills a gap identified in the review, that of wireless-specific hybrid IDS implementations incorporating explainability.

In Chapter Three, system analysis and design were covered; the constraints of current wireless IDS tools were examined, and a comprehensive system requirement specification document was given, which includes eleven functional requirements and six non-functional requirements, as well as a thorough system design, which includes use case, activity diagram, and class diagrams. The three-tier Flask web application architecture was defined, which included the processing tier containing a DataLoader, Preprocessor, DetectionEngine, ExplanationEngine, and DashboardController, and the data tier containing a SQLite database, saved model files, and the AWID3 dataset.

The proposed system was fully implemented and evaluated, which was recorded in Chapter Four. The AWID3 dataset has been preprocessed with a class distribution matching approach over 114 CSV files, resulting in an 80/20 train-test split. A supervised detection model, trained using a Random Forest 100DT classifier, and an unsupervised anomaly detector, trained using only normal traffic data, were used to train the classifiers. An expert system, along with SHAP TreeExplainer, was developed to issue plain English threat explanations for each intrusion detected. All components were integrated and tested using eight withheld test files in the Flask dashboard. The system has successfully realized perfect recall for all three attack classes on real AWID3 test data, and an average accuracy of 97.87% on average across five synthetic wireless environment evaluations. The implemented system achieved all five project objectives.

5.3 Conclusion

The primary challenge in this project was to develop and deploy a hybrid supervised and unsupervised intrusion detection system for wireless networks to overcome three key shortcomings in the literature: (1) lack of wireless-specific benchmark evaluation, (2) the absence of integrated explainability mechanisms in wireless IDS implementations, and (3) the absence of a hybrid supervised and unsupervised detection architecture for IEEE 802.11 environments. The system implemented directly tackles the three gaps by leveraging the AWID3 wireless dataset, two model detection engines (Random Forest and Isolation Forest), and the rule-based expert system with the help of SHAP explainability.

In terms of the five specific objectives, the following conclusions are drawn. The first goal - obtaining and preprocessing the AWID3 data set - was completely met by a rigorous preprocessing pipeline, including train-test separation, class distribution matching, missing value imputation, feature scaling, and label encoding, all performed without relying on information that might have leaked from the test set to the training set. This second goal, creating a hybrid IDS for deauthentication, disassociation, and rogue access point attacks, was successfully met by using a Random Forest (RF) classifier with 99.44% accuracy and a structurally anomalous traffic detector based on Isolation Forest (IF), operating independently and able to identify structurally anomalous traffic for all of the test sessions. The third goal — to embed a rule-based expert system and SHAP explainability engine — was completely met, and the system yields structured plain-English explanations about the attacks for every intrusion detected, addressing attack mechanism, potential impact, and recommended response actions. The fourth goal — develop a near real-time monitoring dashboard using Flask — was completely met, and it was able to successfully stream detection results, show attack type distributions, calculate session metrics in real time, and store

detection results in a SQLite database. The 5th objective – Assess the performance of models using standard measures – was successfully met with 313,193 test records on eight held-out files and 50,000 synthetic evaluation records on five wireless network environments.

The model's performance on the real AWID3 held-out test data set is consistent with the published benchmark results for the AWID data set (Abdulhammed et al., as cited in Ozkan-Okay et al., 2021), where the same model (Random Forest) achieved 99.64% accuracy on the AWID data set, validating the model's performance against the established literature standard. The 5 experiments, each conducted in a synthetic environment, yielded an average accuracy of 97.87%, showing that the system has a strong detection capability in environments with greater variation in features than the controlled AWID3 laboratory environment, thus demonstrating the generalisation from the specific training set to other environments. The zero missed attack (100% recall) occurred in all three attack classes, indicating that the system meets its primary security requirement of catching every real intrusion, which is most important for any IDS deployment (Vinayakumar et al., 2019; Sowmya et al., 2023).

This project is aware of one major limitation, namely, the AWID3 dataset is controlled. AWID3's attack files are highly inter-session consistent and are produced as a laboratory-captured benchmark, which helps to maintain high classification metrics. This is in line with the well-known research results on the dataset, but may not be interpreted as a metric of performance for heterogeneous real-world wireless traffic. While the synthetic evaluation somewhat mitigates this limitation, it does not completely replace the evaluation on a true diversity of live wireless captures. This limitation can be overcome by future implementations via live traffic capture and evaluation on more heterogeneous wireless network datasets.

5.4 Recommendations

From the results of this project, the following recommendations are made for further research, development, and implementation of AI-based wireless Intrusion Detection systems:

Live Traffic Capture Integration: The most significant improvement to this system would be live IEEE 802.11 packet capture (e.g., Scapy or PyShark). Without pre-captured CSV files, there would be no need for real-time packet capture, and true real-time intrusion detection would be possible. In theory, it can be done in Flask, and a detection pipeline already exists to do so, albeit with a relatively small change in the DataLoader component.

Evaluation over Other Wireless Datasets: Future work should include evaluating the trained model with other wireless datasets beyond AWID3, such as AWID2, captures from different hardware environments via the WLAN, and more recent datasets to capture 802.11ac and 802.11ax (Wi-Fi 6) wireless traffic. The cross-dataset evaluations would provide more insight into the capacity of the model to transfer to other datasets, and the kinds of distributions that the current model is not capable of handling.

Deep Learning Model Integration: The interpretability of Random Forest and good benchmark results led to the selection of this model as the additional model for implementation; deep learning models like LSTM or CNN for pattern recognition in time series. Deep learning architectures are able to perform competitive accuracy on IDS tasks and can be able to include sequential dependence of wireless traffic that ensemble methods cannot. The deep learning architecture can achieve competitive accuracy in performing the IDS task and can be equipped to incorporate sequential dependence of wireless traffic that can be not available for ensemble methods, as demonstrated by Ikeri (2025) and Vinayakumar et al. (2019).

Extended Attack Coverage: The existing system is focused on three types of attacks in AWID3 (Extended Attack Coverage). Future versions should include further types of wireless attacks, including beacon flooding, PMKID attacks, WPA3 downgrade attacks, and channel-based jamming, which comprise a large portion of the wireless attacks observed in the wild, but were absent from the AWID3 dataset. Additional training data would be required for the introduction of new attack classes, which could be acquired in a controlled testbed scenario in the wireless domain, or adversarial simulated scenarios.

Multi-User Authentication and Role Management: The current system is not authenticated, which is OK for a prototype but not for production. A future iteration should have role-based access control (RBAC), which would allow for multiple network administrators of varying access levels to log on to the dashboard, view alerts, and manage detection sessions. This will align with enterprise security operations center (SOC) deployment needs.

References

Acka, B. C., Billong, S. C., Djotio, T., & Nlong, J. M. (2025). AI-Based Intrusion Detection and Prevention System for Securing Wireless Communication Networks. International Journal of Safety and Security Engineering, 15(1), 117–126.

Amachaghi, E. (2025). AI-based IDPS for Open RAN 5G Wireless Networks [PhD Thesis]. University of South Wales.

Dash, B., Sharma, P., Ali, A., Gour, M., Shaikh, A., & Singh, A. (2022). AI-Based Intrusion Detection Systems to Secure IoT with Edge Computing Environments: A Comprehensive Review. 5th International Conference on Data Science and Information Technology (DSIT 2022). https://doi.org/10.1145/3544109.3544110

Fares, H., Vibhute, A. D., Mouniane, Y., & Bouijij, H. (2025). Intrusion Detection in Wireless Sensor Networks using Machine Learning. Procedia Computer Science, 252, 912–921.

Ghadami, S. (2025). An Advanced Intrusion Detection Framework for IoT Environments Using Game-Theory-Based GANs, Hybrid Feature Selection, and Parallel CNN-LSTM. Scientific Reports, 15, 38156. https://doi.org/10.1038/s41598-025-22074-3

Ikeri, D. (2025). AI-Powered Intrusion Detection Systems: A Comparative Study of Machine Learning Models for Network Security [Thesis]. JAMK University of Applied Sciences.

Masengo Wa Umba, A., Abu-Mahfouz, A. M., Ramotsoela, D., & Hancke, G. (2022). Towards Artificial Intelligence-Driven Intrusion Detection for Software-Defined Wireless Sensor Networks. International Journal of Environmental Research and Public Health, 19(5), 3670. https://doi.org/10.3390/ijerph19053670

Ozkan-Okay, M., Samet, R., Aslan, O., & Gupta, D. (2021). A Comprehensive Systematic Literature Review on Intrusion Detection Systems. IEEE Access, 9, 157727–157760. https://doi.org/10.1109/ACCESS.2021.3129336

Pant, B., Jaiswal, A., Mankar, A., Singh, R., & Rajput, S. (2023). A Lightweight AI-Enhanced Intrusion Detection System for WBANs. IEEE Access, 11(1), 1–15.

Saranya, T., Sridevi, S., Deisy, C., Chung, T. D., & Khan, M. K. A. A. (2020). Performance Analysis of Machine Learning Algorithms in Intrusion Detection System: A Review. Procedia Computer Science, 171, 1251–1260. https://doi.org/10.1016/j.procs.2020.04.133

Sowmya, T., & Anita Mary, 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

Vanhoef, M., & Piessens, F. (2017). Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS 2017), 1313–1328. https://doi.org/10.1145/3133956.3134027

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep Learning Approach for Intelligent Intrusion Detection System. IEEE Access, 7, 41525–41550. https://doi.org/10.1109/ACCESS.2019.2895334

Wang, S., Yao, Y., Han, T., & Zhao, S. (2022). An AI-Powered Network Threat Detection System. IEEE Access, 10, 54029–54043. https://doi.org/10.1109/ACCESS.2022.3175929

Yellepeddi, S. D., Kiran, P. R., Reddy, M. A., Dhanalakshmi, R., Pooja, D., & Suresh, M. (2024). Analysis of Artificial Intelligence Integrated IDS for the Cyber Threats in the Real-Time

Networks. 2024 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT), 947–953.