ADDIS ABABA UNIVERSITY

COLLEGE OF NATURAL AND COMPUTATIONAL SCIENCES

Jamming Attack Detection and Classification using Exponentially Weighted Moving Average and Random Forest in Wireless Sensor Networks

Alemayehu Ebissa

A Thesis Submitted to the Department of Computer Science in Partial Fulfillment for the Degree of Master of Science in Computer Science

Addis Ababa, Ethiopia

October 2024

Addis Ababa University

College of Natural and Computational Sciences

Alemayehu Ebissa

Advisor: Mulugeta Libsie (PhD)

This is to certify that the thesis prepared by *Alemayehu Ebissa*, titled: *Jamming Attack Detection and Classification using Exponentially Weighted Moving Average and Random Forest in Wireless Sensor Networks* and submitted in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the Examining Committee:

Name	Signature	Date
Advisor: Dr. Mulugeta Libsie	_____	_____
Examiner: _____	_____	_____
Examiner**:** _____	_____	_____

Abstract

Wireless sensor networks are widely used in environmental monitoring, industrial automation, healthcare, and smart cities for data collection, real-time monitoring, and automated decision-making. These networks, consisting of randomly distributed autonomous nodes, are vulnerable to jamming attacks where malicious entities disrupt network transmissions by emitting interfering signals. Existing detection methods typically rely on either statistical or machine learning-based approaches, each with significant limitations: statistical-based methods are prone to high false alarm rates, while machine learning-based methods impose computational overhead on resource-constrained nodes. To address these limitations, this thesis presents a two-level jamming attack detection and classification method that combines the strengths of both approaches. The method integrates an Exponentially Weighted Moving Average (EWMA) for lightweight detection with a Random Forest classifier for accurate jamming attack classification. The approach begins with feature selection, utilizing key features such as the Received Signal Strength Indicator (RSSI) and Packet Error Rate (PER), which can be easily obtained without adding significant overhead on sensor nodes. The method consists of a training phase and a testing phase. In the training phase, the dataset is processed through the EWMA computation to smooth the time-series data, followed by threshold calculation. The EWMA-smoothed data is then used to train the Random Forest classifier. In the testing phase, the testing dataset also passes through the EWMA computation, and the EWMA-based jamming detection determines if a jamming attack is occurring by comparing against a predefined threshold. Once potential jamming is detected, the system transitions into the classification of the three jamming types: constant, periodic, or reactive jamming. Experimental evaluation demonstrates that our method achieves a 99.91% detection rate and 99.26% accuracy in jamming classification. These results show significant improvements over existing methods, particularly in reducing false positives while maintaining high detection accuracy.

Keywords: Exponentially Weighted Moving Average, Jamming Detection, Radio Jamming, Random Forest, Wireless Sensor Network.

Acknowledgments

First and foremost, I would like to praise and thank God, the Almighty, for His countless blessings throughout this journey. Without His grace and strength, I would not have been able to overcome the challenges and successfully complete this thesis.

I would also like to extend my heartfelt thanks to my advisor, Dr. Mulugeta Libsie, for his invaluable guidance, insightful feedback, and support throughout this research. His expert advice and encouragement have been significant in helping me organize and refine this thesis. His attention to detail has greatly enriched my work, ensuring that the document is both coherent and well-structured.

Lastly, I am deeply thankful to my family for their unconditional love, patience, and support. Their constant encouragement and belief in my abilities have given me the strength to persevere through the challenges I encountered. I would not have been able to achieve this milestone without their unwavering presence and motivation.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Acronyms and Abbreviations

AM	Amplitude Modulation
ANN	Artificial Neural Network
BER	Bit Error Rate
CH	Cluster Head
ECA	Energy Consumption Amount
EVM	Error Vector Magnitude
EWMA	Exponentially Weighted Moving Average
FEC	Forward Error Correction
FM	Frequency Modulation
GBDT	Gradient Boosted Decision Tree
IoT	Internet of Things
KNN	K-Nearest Neighbors
LCL	Lower Control Limit
LEACH	Low Energy Adaptive Clustering Hierarchy
MLP	Multi-Layer Perception
PDR	Packet Delivery Ratio
PER	Packet Error Rate
PM	Phase Modulation
PSR	Packet Sending Ratio
RF	Radio Frequency
RSSI	Received Signal Strength Indicator
SINR	Signal to Interference plus Noise Ratio
SVM	Support Vector Machine
TCP	Transmission Control Protocol
UCL	Upper Control Limit
WSN	Wireless Sensor Network

Chapter One: Introduction

1.1 Background

Wireless Sensor Networks (WSNs) are networks formed by a vast number of spatially distributed autonomous nodes, each equipped with sensors, embedded processors, and low-power radios to enable wireless communication among themselves and with a central base station [1]. These networks serve as a crucial infrastructure for monitoring and collecting data from the physical world in real-time. WSNs are extensively used in various industries like military, industry, health, smart cities, etc. [2].

WSNs are composed of sensor nodes and a base station. The base station is tasked with the processing and analysis of the collected data. The positioning of the base station is essential as it can affect key network performance measures such as energy efficiency, data throughput, and latency [3]. The sensor nodes are responsible for gathering data from the environment. Sensor nodes are often situated in remote, and unmonitored environments. WSNs operate within a set of constraints and are subject to various limitations. These include limited computational resources, confined storage space, restricted bandwidth, short communication distance, slower data transmission speed, and limited battery life [4]. These factors, combined with the inherent vulnerability of the wireless medium, render sensor nodes susceptible to various security threats [5]. These threats, which can target various layers of the network, include jamming attacks, sinkhole attacks, session hijacking, desynchronization, and others. Among these, one of the most prevalent security threats is the radio frequency jamming attack at the physical layer, where attackers emit radio signals at the same frequency to disrupt network communication [6].

Jamming attacks in WSNs are carried out by rogue nodes within the network, intending to disrupt or interfere with the transmission and reception of authentic wireless signals exchanged among sensor nodes [1]. Jamming differs from regular noise or interference in that it results from the intentional use of wireless signals to degrade network performance, while interference constitutes unintentional forms of noise that disrupt the performance of WSNs [7]. Various jamming attack types have been identified in the literature, including constant, reactive, and periodic jamming [8]. In constant jamming, the adversary persistently transmits radio frequencies to congest the channel, preventing legitimate nodes from successfully sending or receiving data. Reactive jamming occurs when the attacker monitors the channel and transmits disruptive signals only upon detecting legitimate transmissions, thereby

interfering directly with active communications. In periodic jamming, the attacker unpredictably switches between jamming and sleeping modes, making detection and avoidance more challenging.

In order to sustain the uninterrupted operation of WSN, it is essential to achieve accurate detection of jamming attacks [2]. Detecting jamming attack has been an ongoing research area in wireless networks over the last decade. Approaches for detecting and classifying jamming can be broadly categorized into centralized and distributed methods [1]. In Centralized approach, only a single node is used to detect the attack, so there is a possibility to increase false positives. In distributed approach, detection of attacks is performed at multiple levels of the network, has more communication overhead and consume more energy. Various strategies have been suggested by the research community to tackle the problem of jamming attacks in WSNs.

Statistical-based methods are widely used for detecting jamming attacks in WSNs. These methods are advantageous because they can detect a jamming attack event by using a minimum number of metrics [2, 5]. Time series analysis is one such statistical method that has been used to detect jamming attacks in wireless networks. This method models the network measurements taken over time as time series and employs a sequential change point detection algorithm to detect the change of state in the time series, which is an indicator of change in the network state [9]. Another statistical method is the exponentially weighted moving average (EWMA), which can detect anomalous changes in the intensity of a jamming attack event by using packet inter-arrival times [5]. These methods can efficiently and accurately detect jamming attacks in WSNs with little or no overhead, making them ideal for resource-constrained sensor nodes [5]. However, statistical-based methods have notable limitations, particularly in multi-class classification tasks. They are inherently less effective at distinguishing between various types of jamming attacks, as they often require predefined thresholds for detection, which must be manually set. It is important to classify jamming attack to take specific countermeasure [13]. Additionally, statistical methods are susceptible to high false-positive rates, as they may interpret benign anomalies such as congested links or weak signal links as jamming attacks, leading to unnecessary alerts that increase resource consumption and reduce network efficiency [8]. Reducing these false positives is essential, as excessive false alarms can degrade performance and hinder the deployment of precise countermeasures.

Several recent studies have also proposed various machine learning-based methods for detecting jamming attacks in wireless network including Random Forest (RF), Support Vector Machines (SVM), Artificial Neural Network (ANN), Multilayer Perception (MLP), Gradient Boosted Decision Tree

(GBDT) and k-Nearest Neighbors (KNN) [10-15]. The above-mentioned jamming attack detection methods can detect and classify jamming attacks with high accuracy. However, implementing such techniques on sensor nodes poses challenges due to the nodes' limited processing capabilities and finite battery life [8]. Recently, the emergence of TinyML has enabled the consideration of machine learning approaches for anomaly detection in resource-constrained environments. TinyML is a field focused on deploying machine learning models on resource-constrained devices. TinyML facilitates the execution of machine learning tasks on edge devices, allowing models to run locally on the sensor node [75]. While TinyML allows for machine learning-based detection on the node itself, the computational overhead remains higher than that of statistical-based methods, which may still be a limitation for resource-limited sensor networks [76].

This research aims to design and evaluate a jamming attack detection and classification method in WSNs using EWMA and Random Forest to address each method’s limitations while leveraging their combined strengths.

1.2 Motivation

WSNs have become integral to many critical applications, particularly in the military sector where continuous and uncompromised functionality is crucial. However, WSNs face significant challenges, one of the most pressing being the detection of jamming attacks. These attacks, which involve malicious interference with wireless communication, are both easy to launch and a prevalent threat in wireless technology.

Recent incidents of jamming attacks highlight the potential consequences of undetected jamming attacks. To address these evolving threats and unforeseen incidents, accurate detection of jamming attacks plays a vital role in defending against intruders. This research aims to address this imperative need by designing and evaluating an accurate detection method that also minimizes the false positive.

1.3 Statement of the Problem

WSNs are susceptible to various attacks, with jamming attack being one of the most prominent security threats. Attackers initiate jamming attack to disrupt data collection and transmission or to drain node resources. This type of attack is relatively easy to launch, as it does not require an understanding of the network's communication protocols and adherence to the MAC protocol, making it a prevalent threat.

Many countermeasures have been proposed to address jamming attack [8]; however, not all of these countermeasures are effective against every type of jamming attack. In many cases, the best approach is to detect the jammer [10]. Detecting radio jamming attacks is particularly challenging, as it requires distinguishing between legitimate causes of connectivity issues, such as weak link, congested link or environmental interference, and intentional disruptions caused by adversaries. This complexity highlights the importance of an effective jamming attack detection and classification method in securing WSNs.

Recent studies have proposed various approaches to detect jamming attacks in WSNs. In the literature, statistical-based methods have been shown to be particularly efficient for resource-constrained environments [2, 5, 9]. Osaniye *et al.* [5] applied the Exponentially Weighted Moving Average (EWMA), a statistical control technique, to detect jamming attacks. The authors reported that their approach achieved up to 100% detection accuracy after the 21st packet. However, a notable limitation of this study is its lack of a clear discussion on the false positive rate. Although the method claims high accuracy, it relies on a single feature, packet inter-arrival time (IAT), which does not guarantee that variations in this feature are solely due to jamming. Other factors, such as network congestion or weak signal links, can produce similar effects [10]. Additionally, statistical methods, including EWMA, are often prone to high false positive rates due to manual threshold tuning [70]. In a related study, one of the pioneering works in jamming attack detection by Xu *et al.* [7] demonstrated that Packet Delivery Ratio (PDR) is an effective metric for detecting jamming attacks. However, their experimental evaluation demonstrated that PDR alone cannot distinguish whether a low PDR is caused by natural factors, such as poor link quality, or other issues. It also revealed that no single metric is sufficient to detect all types of jamming attacks. To address this limitation and improve detection accuracy, they proposed two strategies that enhance PDR-based detection. These strategies integrate signal strength measurements with PDR to perform consistency checks, helping to identify whether low PDRs result from natural causes or radio interference. This underscores the importance of utilizing multiple features for more accurate detection of jamming attacks.

In wireless networks, machine learning-based methods are also widely used for detecting and classifying jamming attacks. Several recent studies show that machine learning-based methods can classify jamming attacks with high accuracy. Arjoune *et al.* [15] demonstrated that Random Forest could achieve an accuracy of 96.6%. Testi *et al.* [12] proposed a single-hidden-layer feed-forward Neural

Network algorithm that achieved up to 99% accuracy in detecting and classifying jamming attacks. Upadhya *et al.* [11] achieved 99.01% accuracy in simulations using Random Forest boosted with Real AdaBoost. Most of these studies, however, are proposed for general wireless networks without considering resource constraints, which is a critical aspect in WSNs. Related researches on machine learning-based jamming detection and classification highlights that Random Forest often achieves higher accuracy compared to other machine learning algorithms. Random Forest is an ensemble learning based classifier that uses the Bagging technique, where several base models are trained independently on bootstrapped samples of the training data [72]. These models' predictions are then combined through averaging or voting methods, enhancing the overall model's accuracy and robustness. The Random Forest algorithm was first introduced by Ho in 1995 [73], utilizing the random subspace method. Later, in 2001, Breiman [74] extended this approach to create the modern version of the algorithm. Random Forest consists of multiple decision trees, each trained on a randomly selected subset of features and data points. By aggregating the predictions of these trees through majority voting, the algorithm achieves improved performance and reduces the risk of overfitting. Its ease of implementation, fast execution, and resistance to overfitting have made Random Forest widely applicable in various fields, including anomaly detection, natural language processing, image recognition, and computer vision [72]. However, despite its demonstrated effectiveness, its application to jamming attack detection and classification in WSNs remains limited.

In response to the constraints of WSNs, recent studies have proposed TinyML, a set of techniques that enables machine learning algorithms to run on resource-constrained nodes. TinyML brings machine learning capabilities to the edge by running models directly on the sensor nodes, thereby reducing the need for data transmission and centralized processing [75]. TinyML typically leverages methods like model compression, quantization, pruning and knowledge distillation to minimize the resource demands of machine learning models while maintaining their accuracy [76]. These strategies make it feasible to deploy machine learning models, including Random Forest, on nodes with limited memory and processing power [75]. While Random Forest typically requires more memory and processing power than some simpler algorithms, it remains one of the most suitable machine learning methods for TinyML due to its ability to be pruned effectively. Pruning can significantly reduce computational overhead while preserving accuracy by removing branches that contribute little to the overall prediction. Additionally, limiting the number of trees and adjusting the depth of each tree can make the model more suitable for deployment on resource-constrained devices [77].

Although TinyML enables machine learning models to run directly on sensor nodes, it still incurs a higher computational overhead compared to traditional methods [78], such as statistical-based approaches, which are less resource-intensive but more prone to high false positive rates [6]. In operational environments where jamming attacks occur sporadically, the continuous execution of machine learning models may lead to unnecessary consumption of node resources. This study addresses this gap in jamming attack detection by proposing a two-level system utilizing EWMA and a Random Forest classifier. The EWMA-based jamming detection, which serves as the lightweight first stage of the proposed method, detects jamming by giving more weight to recent observations while gradually reducing the influence of older data, allowing it to identify sudden shifts in network behavior indicative of jamming. This approach is well-suited to the resource limitations of WSN nodes [5]. In the second stage, Random Forest classifier, will be employed for accurate classification of the detected jamming attack and reduce false positive rate that are resulted from EWMA-based jamming detection.

Furthermore, while most existing studies focus primarily on detecting jamming attacks, it is essential to highlight the importance of classifying these attacks. Classifying jamming attacks is crucial because it enables the implementation of appropriate countermeasures that are both effective and efficient for preventing and recovering from such attacks [13]. This classification plays a key role in ensuring a targeted response to different types of jamming, optimizing the overall security of the network. Another gap in the literature is that many proposed solutions are evaluated using simulated data. However, simulation data often fails to account for real-world variables, such as unintentional interference and environmental noise, which can significantly impact detection rates and the accuracy of these systems in actual applications. In contrast, this study evaluates the proposed method using a dataset collected from a real-world experiment conducted by Jacovic *et al.* [13]. In their study, three types of jamming attacks: constant, reactive, and periodic were launched. They also considered factors like weak links and network congestion, which are common in real-world WSNs. This realistic dataset allows for a more accurate evaluation of the proposed method's performance, ensuring its effectiveness in real-world deployments.

The focus of this research is on improving jamming attack detection and classification accuracy while minimizing false positive rate.

1.4 Objectives

General Objective

The general objective of this research is to design and evaluate a jamming attack detection and classification method using EWMA and Random Forest in WSN.

Specific Objectives

- To conduct a comprehensive review of the current state-of-the-art in jamming attack detection and classification methods within WSN.
- To collect and acquire dataset for training the detection and classification model.
- To design a jamming attack detection and classification method.
- To evaluate the proposed method performance.

1.5 Methods

In conducting this research to achieve the stated objectives, the research work begins with literature review to gather, analyze, and synthesize information from existing papers and electronic sources related to jamming attacks in WSNs. This review helps in identifying relevant research gap, shape the research objectives and techniques, particularly related to jamming attack detection. By examining various approaches to jamming detection, a deeper understanding of the field is established.

The research methodology includes a two-level detection method. This method employs an EWMA-based jamming detection for initial jamming detection, followed by a Random Forest classifier for jamming classification. This layered approach enhances both detection accuracy and classification accuracy while addressing the resource constraints of sensor nodes.

For data collection, this research utilizes an existing dataset previously used by other researchers to evaluate their jamming detection methods. Finally, the experimentation and evaluation phase involve testing the proposed solution using a testing dataset. The performance of the solution is assessed through key metrics, including accuracy, precision, recall, and F1-score, ensuring a comprehensive evaluation of the approach's effectiveness in detecting and classifying jamming attacks.

1.6 Scope and Limitations

This research aims to design and evaluate a method for detecting and classifying jamming attacks in WSNs, which is a crucial step in enabling effective countermeasures. The research will use a publicly available dataset, which was obtained from the experimental evaluations conducted by Jacovic *et al.* [13]. Additionally, the proposed method targets a cluster-based sensor node architecture, where cluster heads are responsible for detecting jamming attacks targeting their respective member nodes, while base stations are responsible for detecting jamming attacks targeting the cluster heads. The proposed two-level method first employs an EWMA technique for lightweight jamming detection, followed by a second level for attack classification using the Random Forest classifier. The Random Forest classifier parameters are optimized for model compression and tree pruning, which makes it well-suited for deployment in TinyML framework without significantly compromising classification accuracy. While this design takes into account the constraints of TinyML, the integration and deployment of the proposed method within a TinyML framework are beyond the scope of this research.

It is important to note that this research exclusively addresses the crucial phases of jamming attack detection and classification. The subsequent stages of localization and mitigation, while essential for comprehensive WSN security, are intentionally outside the scope of this study. Furthermore, the focus of this research is primarily on enhancing the accuracy of jamming attack detection and classification by minimizing false positives. The proposed method aims to address the resource limitations of WSN nodes by combining an EWMA-based jamming detection technique, which is lightweight, with a Random Forest classifier, designed to classify the type of jamming attack and reduce false positives. However, this research does not incorporate the implementation of TinyML and does not aim to achieve state-of-the-art results in terms of detection time, energy consumption or memory usage. These considerations are designated as future work.

1.7 Application of Results

WSNs are critical to many applications, including military operations, industrial monitoring, and healthcare systems. The continuous and uninterrupted operation of WSNs is essential to ensure the integrity of these applications. However, WSNs face many challenges, including disruptive jamming attacks, which are increasingly sophisticated.

This research aims to design a two-level detection jamming attack detection and classification method that can detect accurately while minimizing false alarms. The significance of this research is deep, as it reinforces the security of WSNs, thereby strengthening the security of applications critical to national interests, particularly in military operations.

Additionally, the proposed detection system improves the accuracy of detection and classification of jamming attack with minimal overhead, effectively addressing a critical gap in the existing body of knowledge.

1.8 Organization of the Rest of the Thesis

The rest of this thesis is organized into five Chapters. This section gives an overview on the contents of each Chapter.

Chapter Two begins with an overview of WSNs, including their architecture, components, and communication protocols. It then delves into the concept of wireless communication in WSNs and the various types of jamming attacks. Following this, various jamming detection metrics used in the literature are discussed. Finally, the EWMA method is discussed.

Chapter Three reviews existing research on jamming attack detection and classification methods in WSNs. It explores statistical-based methods and machine learning-based methods, highlighting their strengths and limitations.

In Chapter Four, the proposed solution for jamming attack detection and classification in WSNs is presented. The chapter outlines the design considerations for the proposed solution, detailing the integration of the EWMA method and Random Forest classifier. The chapter discusses how these techniques can be combined to improve the accuracy and efficiency of jamming attack detection and classification in WSNs.

Chapter Five provides an overview of the evaluation process conducted to assess the effectiveness and performance of the proposed solution. It discusses the experimental setup, the metrics used for evaluation, and presents the results of the evaluation. The chapter also includes a comparative analysis with existing methods to demonstrate the superiority of the proposed solution.

The final chapter, Chapter Six, of the thesis summarizes the findings of the study and presents conclusions drawn from the research. It discusses the implications of the findings and their significance for security of WSNs. Additionally, the chapter outlines potential directions for future research and improvements to the proposed solution.

Chapter Two: Literature Review

In this literature review chapter, we provide a comprehensive overview of existing research on jamming attack detection in WSNs. It begins by exploring the architecture and characteristics of WSNs, emphasizing their inherent vulnerabilities to jamming attacks. The chapter then examines various jamming attack strategies and their impact on the performance and reliability of WSNs. Additionally, it highlights the key detection metrics employed in identifying jamming attacks. Lastly, the chapter explores the principles of EWMA, highlighting their advantages and limitations in the context of jamming attack detection.

2.1.1 Overview of WSNs

A WSN is a self-organizing network which represents a paradigm shift in data acquisition from geographically dispersed and often challenging environments. These self-organizing networks leverage low-cost, miniature sensor nodes that collaborate to capture a wide range of environmental parameters, including temperature, light, and even chemical signatures [17].

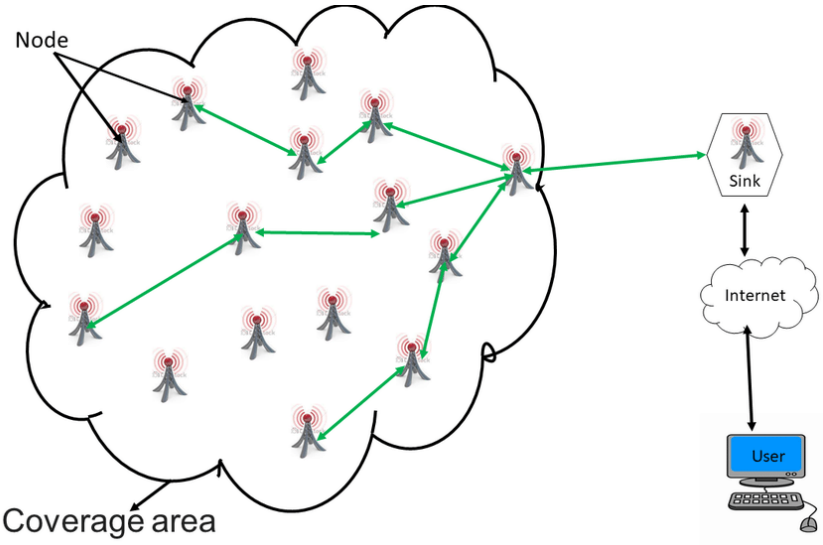


Figure 2.1: General architecture of a WSN [18]

Unlike traditional, centralized networks, WSNs employ a multi-hop communication scheme as shown in Figure 2.1, where sensor nodes cooperatively relay information until it reaches a central collection point called sink node. WSNs must have to be capable of operating in unattended, inaccessible or hazardous

environment, enables data gathering in diverse fields like environmental monitoring, precision agriculture, and remote healthcare [2].

2.1.1 Characteristics of WSNs

Compared to traditional wireless networks, WSNs prioritize fundamentally different application requirements, design goals, and technical specifications. In WSNs, data transmission and communication can be achieved using various routing mechanisms, including multi-hop, single-hop, or cluster-based approaches, depending on the network's design and application requirements. The characteristics of WSNs are summarized as follows [19]:

Decentralized Architecture

The decentralized architecture of WSNs is a fundamental characteristic that distinguishes them from traditional wired networks. In WSNs, sensor nodes operate autonomously and collaboratively, without relying on centralized control or coordination [1, 18]. This decentralized nature empowers WSNs to exhibit adaptive behavior, enabling them to respond dynamically to changes in the environment, node failures, or network disruptions.

Each sensor node in a WSN is equipped with sensing, processing, and communication capabilities, allowing it to perform a variety of tasks independently [2]. Sensing capabilities enable sensor nodes to collect data about the physical environment, such as temperature, humidity, light intensity, or motion. Processing capabilities enable nodes to analyze the collected data, extract relevant information, and make local decisions based on predefined algorithms or rules. Communication capabilities facilitate the exchange of data and control messages between sensor nodes, enabling collaboration and coordination among neighboring nodes [17].

The decentralized nature of WSNs offers several advantages. It enhances network resilience and fault tolerance by eliminating single points of failure and reducing dependency on a central controller [3]. In the event of node failures or communication disruptions, neighboring nodes can autonomously reconfigure themselves and adapt their behavior to maintain network connectivity and functionality. This self-organizing capability is crucial for WSNs to function effectively in dynamic and unpredictable environments, such as those encountered in disaster response scenarios or outdoor monitoring applications [20].

Scalability

The deployment of WSNs in large-scale scenarios presents significant challenges and opportunities due to the expansive coverage and the considerable number of sensor nodes involved. These deployments often span vast geographical areas, ranging from remote outdoor environments to densely populated urban regions. In such expansive deployments, the scalability of the network becomes paramount, as it directly impacts network management, data aggregation, and energy efficiency [18].

Managing a large-scale WSN involves coordinating a multitude of sensor nodes spread across the deployment area. This entails tasks such as configuring nodes, maintaining network connectivity, and ensuring data integrity and reliability. As the number of sensor nodes increases, the complexity of network management grows exponentially, requiring efficient algorithms and protocols to automate and streamline administrative tasks [21].

Data aggregation is another critical aspect of large-scale WSNs deployments, as it influences communication overhead, energy consumption, and network throughput. In densely populated deployments with a high density of sensor nodes, the volume of data generated can quickly overwhelm the network, leading to congestion and increased energy consumption [22]. Efficient data aggregation techniques are essential to reduce redundant data transmission, minimize network traffic, and conserve energy. Hierarchical or tree-based data aggregation approaches may be adopted to aggregate data locally at intermediate nodes before forwarding it to the sink node, thereby reducing the communication overhead and energy consumption [17, 22].

Energy Efficiency

Energy efficiency stands as a cornerstone concern in the design and operation of WSNs, primarily due to the inherent constraint posed by the limited energy resources of sensor nodes. Typically powered by batteries, these nodes operate in environments where frequent battery replacement or recharging is impractical or even infeasible [23]. Consequently, the effective management of energy consumption becomes paramount to extend the operational lifespan of the network and ensure its sustained functionality over extended periods.

Given these constraints, researchers have devoted considerable effort to developing and implementing energy conservation techniques tailored specifically to the unique requirements of WSNs [1]. These

techniques encompass a spectrum of approaches aimed at minimizing energy expenditure across various facets of network operation, including communication, computation, and sensing.

One prominent area of focus revolves around the design of energy-efficient communication protocols, which govern how sensor nodes exchange data and control messages within the network [21]. By optimizing transmission schedules, packet sizes, and routing strategies, these protocols strive to reduce the energy overhead associated with wireless communication while maintaining adequate network coverage and connectivity [24]. Examples of such protocols include LEACH (Low Energy Adaptive Clustering Hierarchy) and S-MAC (Sensor-MAC), which leverage strategies like data aggregation, node clustering, and duty cycling to minimize energy consumption [25, 26].

In addition to protocol-level optimizations, energy efficiency is also addressed through advancements in low-power hardware design tailored to the specific requirements of WSNs [27]. This encompasses the development of energy-efficient microcontrollers, radio transceivers, and sensing components optimized for minimal power consumption without compromising performance. Furthermore, innovations in energy harvesting technologies offer promising avenues for replenishing node energy reserves using ambient sources such as solar, thermal, or kinetic energy [28].

Duty cycling represents another essential technique employed to conserve energy in WSNs, whereby sensor nodes alternate between active and sleep states to minimize idle power consumption [29]. By synchronizing their wake-up schedules and operating in a low-power sleep mode when not actively sensing or communicating, nodes can significantly reduce energy expenditure without losing responsiveness or coverage. However, striking an optimal balance between duty cycle duration and responsiveness is crucial to ensure timely data collection and maintain network performance [30].

Resource Constraints

Sensor nodes in WSNs are subject to various resource limitations, including processing capabilities, memory capacity, and communication bandwidth [31]. These constraints pose significant challenges to the efficient operation of WSNs, necessitating the development of specialized techniques and algorithms to overcome them.

Due to the limited processing capabilities of sensor nodes, resource-constrained devices must perform data processing tasks efficiently to conserve energy and optimize network performance. Localized data

processing and aggregation techniques are employed to minimize the amount of data transmitted over the network, reducing communication overhead and conserving energy [24, 26]. By processing data locally at the source or intermediate nodes, unnecessary transmissions of raw sensor data can be avoided, resulting in significant energy savings [25].

Moreover, the limited memory capacity of sensor nodes imposes constraints on the amount of data that can be stored and processed locally. Therefore, efficient data storage and management techniques are essential to maximize the utilization of available memory resources [32]. Compression algorithms, data summarization techniques, and adaptive sampling strategies are employed to reduce the size of data stored and processed by sensor nodes, allowing them to operate within their memory constraints [33].

Communication bandwidth is another critical resource constraint in WSNs, as sensor nodes typically operate in bandwidth limited wireless channels. To optimize communication efficiency and minimize energy consumption, data transmission protocols and routing algorithms must be carefully designed to utilize available bandwidth effectively [24]. Techniques such as data aggregation, data fusion, and opportunistic routing are employed to reduce the number of transmissions and maximize the throughput of the network [26].

Advanced algorithms and optimization techniques play a crucial role in addressing the resource constraints of sensor nodes and ensuring optimal network performance in WSNs [34]. Through the use of machine learning algorithms, optimization models, and heuristic methods, resources can be distributed in a manner that maximizes efficiency [35]. These tools also enable intelligent task scheduling and the dynamic adjustment of network parameters to accommodate fluctuating environmental conditions. By harnessing the power of these techniques, WSNs can optimize the use of their limited resources, thus extending the operational lifespan of their sensor nodes [36].

2.1.2 Architecture of WSN

The architecture of WSN is a fundamental aspect that determines the efficiency, scalability, and reliability of the network in fulfilling its intended purpose. It includes Sensor Nodes, Sink Nodes, Communication Infrastructure, and Network Architecture, each crucial for the network's performance and reliability. In this section, we will discuss sensor nodes' roles in data sensing and communication, the central coordinating functions of sink nodes, and the impact of various communication protocols on

network efficiency. Finally, we will explore different network topologies, such as hierarchical and flat architectures, and their implications for scalability and energy efficiency.

Sensor Nodes

Sensor nodes are the basic building blocks of WSNs, responsible for sensing environmental parameters, processing data, and wirelessly communicating with other nodes in the network. Each sensor node is equipped with one or more sensors to measure physical phenomena such as temperature, humidity, light intensity, pressure, and motion [31]. Additionally, sensor nodes contain processing units, memory, and communication interfaces for data transmission as shown in Figure 2.2. The design of sensor nodes is typically optimized for low power consumption to prolong battery life and ensure long-term operation in resource-constrained environments [36].

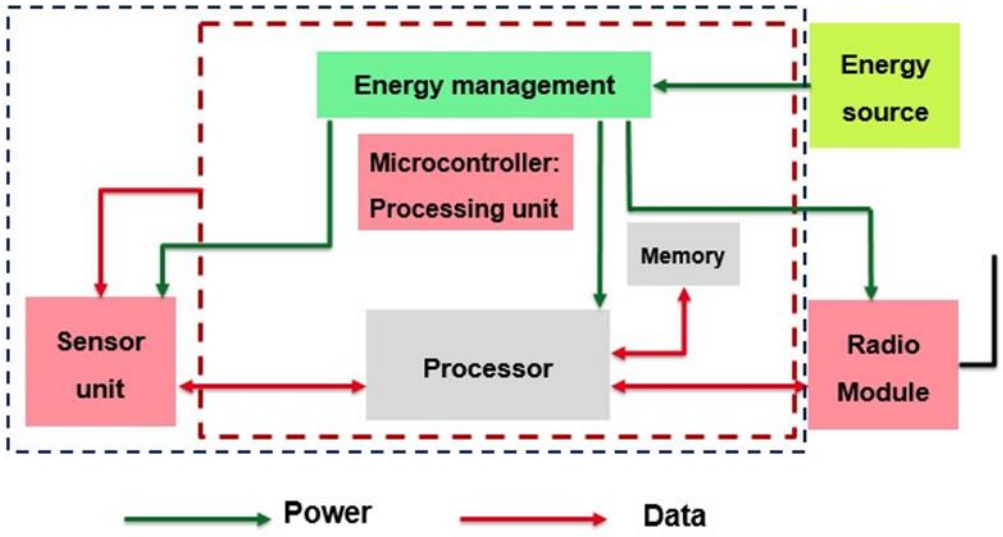


Figure 2.2: The typical architecture of a sensor node used in WSNs [37]

The architectures of sensor nodes vary depending on the specific application requirements and environmental conditions. In an environmental monitoring application, sensor nodes may be designed to withstand harsh weather conditions and operate outdoors for extended periods [38]. In a healthcare monitoring application, sensor nodes may be embedded in wearable devices or implanted in the human body, requiring miniaturization and biocompatibility [39]. The choice of sensors and sensing modalities also depends on the target application, with different sensors available for measuring temperature, humidity, light, sound, vibration, and chemical substances [40].

Sink Nodes

Sink nodes, also known as base stations or gateway nodes, serve as the central coordinators of the WSN, responsible for collecting data from sensor nodes, processing and aggregating the data, and forwarding it to external networks or user [19]. Sink nodes are strategically positioned within the monitoring area to facilitate efficient data collection and dissemination across the network. They are typically equipped with more powerful processors, larger memory, and more reliable communication interfaces compared to sensor nodes, enabling them to handle the computational and communication tasks required for data aggregation and network management [41].

Sink nodes play a critical role in ensuring the reliability and scalability of WSNs. They are responsible for establishing and maintaining communication links with sensor nodes, coordinating data collection and aggregation, and managing network resources [22]. Sink nodes are typically deployed in fixed locations, such as central monitoring stations. However, in some specialized applications, mobile platforms, such as unmanned aerial vehicles (UAVs) or mobile vehicles, can be used to adapt to dynamic environmental conditions and improve data collection efficiency [20]. The placement and configuration of sink nodes significantly impacts the performance and energy efficiency of the WSN [42]. By strategically placing sink nodes and designing efficient routing protocols, researchers can minimize communication overhead, reduce energy consumption, and extend the network lifetime [41, 43].

Communication Infrastructure

WSNs rely heavily on their communication infrastructure to enable sensor nodes to collaborate and transmit data effectively. This infrastructure is governed by communication protocols, which establish the rules and mechanisms for various crucial functions within the network. These functions include channel access, data transmission, error detection and correction, and overall network management [44].

The selection of an appropriate communication protocol significantly impacts the overall performance and efficiency of the WSN. Choosing the right protocol involves considering numerous factors that affect the network's functionality [45]. One key consideration is the communication range, which refers to the required distance for sensor nodes to communicate with each other and with the sink node. Protocols like Zigbee and Bluetooth are suitable for short-range communication, while LoRaWAN offers long-range capabilities. Another important factor is the data rate, which pertains to the volume of data that needs to be transmitted within a given time frame. In situations that demand high data rates, Wi-Fi is an ideal

choice, while Zigbee prioritizes low power consumption over achieving high data rates. Power consumption is also a critical consideration, as sensor nodes usually have limited battery life. Therefore, protocols that consume less power are preferred to extend the network's lifetime. Examples of such protocols include Zigbee and LoRaWAN, both of which are designed for low-power operation. Finally, network topology, or the physical layout of sensor nodes within the network, plays a role in protocol selection. Some protocols are tailored for clustered WSNs and require specific features to facilitate hierarchical communication structures.

Zigbee, Bluetooth, Wi-Fi, and LoRaWAN are among the most commonly used protocols in WSNs. Zigbee is an ideal choice for environments where energy efficiency is crucial and sensor nodes are closely packed, making it perfect for applications like home automation and industrial monitoring that require regular data exchange of moderate size [46]. Bluetooth, on the other hand, while similar to Zigbee in terms of low-power, short-range communication, stands out with its higher data transfer speed, making it suitable for personal area networks and wearable health devices within WSNs [47]. Wi-Fi, however, prioritizes high data speeds and long-distance communication, making it suitable for situations that require quick data transmission over large areas, such as smart city projects and infrastructure monitoring.

It is important to note, though, that Wi-Fi consumes significantly more power than Zigbee or Bluetooth [48]. On the other hand, LoRaWAN provides a unique combination of long-distance communication and low power usage, making it especially useful for wide-ranging applications in sectors like agriculture and environmental monitoring where sensor nodes are geographically spread out. These protocols, each with their unique strengths, play a crucial role in the efficient functioning of WSNs [49].

Network Architecture

WSNs can be organized into various network architecture based on the arrangement of sensor nodes and sink nodes in the network. Each topology offers unique advantages and trade-offs in terms of scalability, fault tolerance, energy efficiency, and deployment complexity. The network architecture commonly employed in WSNs includes hierarchical architecture and flat architecture [50].

Hierarchical architectures introduce a layered structure, dividing nodes into clusters with designated cluster heads (CHs). CHs aggregate data from their member nodes and forward it to the sink, reducing the number of hops required for data transmission. This approach offers significant energy savings and

improved scalability compared to flat networks. Research by Guleria and Verma [51] highlights the benefits of hierarchical architectures, including efficient data aggregation, reduced latency, and improved network lifetime. However, challenges exist in selecting optimal CHs, maintaining cluster stability, and ensuring efficient data aggregation within clusters. Additionally, hierarchical protocols require more complex algorithms and introduce additional overhead compared to flat architectures. While the basic principle of hierarchical architectures involves cluster heads and sensor nodes, several variations exist within this category, each offering distinct advantages and considerations. One of the most commonly deployed is a cluster-based architecture, organizes sensor nodes into clusters, where each cluster is managed by a cluster head responsible for aggregating data from member nodes and forwarding it to the sink node [52]. In the context of cluster-based WSNs, the network architecture can be viewed as a two-layered hierarchical system as shown in Figure 2.3. Cluster heads, distinguished by their expanded functionalities, occupy the upper layer, while standard sensor nodes operate in the lower layer. These cluster heads typically possess greater capabilities compared to regular nodes, encompassing enhanced radio communication range, processing power, sensing abilities, and power supply capacity [53].

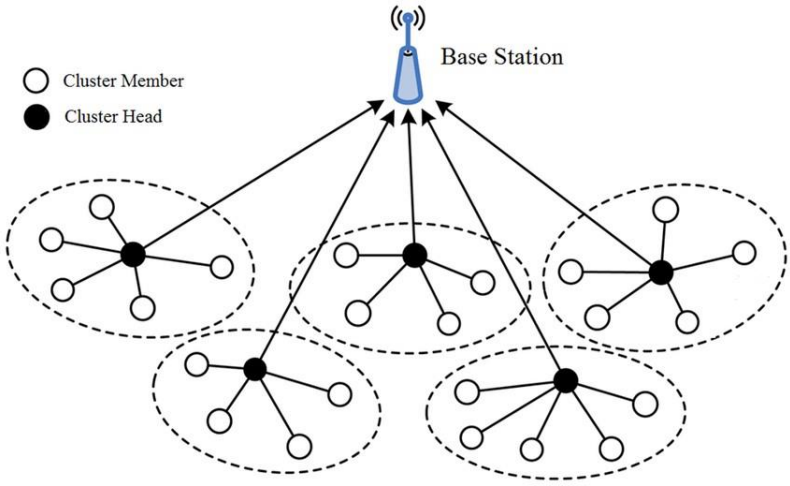


Figure 2.3: Cluster-based WSN [18]

Cluster-based WSNs are well suited for large-scale deployments with distributed sensor nodes, such as environmental monitoring and precision agriculture. However, they may suffer from increased latency and complexity in cluster formation and maintenance [54].

In a flat architecture, all sensor nodes communicate directly with the sink or base station. This approach is straightforward and suitable for small-scale deployments where data latency i`s not a critical concern. Flat routing protocols, such as flooding or SPIN, are employed, where each node forwards received data

to all its neighbors [55]. Flat architectures offer advantages in terms of simplicity and adaptability. However, they suffer from scalability limitations as the number of nodes increases. Data packets need to traverse multiple hops, leading to higher energy consumption and increased latency. Additionally, flat networks are more susceptible to congestion and single points of failure due to the reliance on the sink nodes [54].

2.1.3 Applications of WSN

WSNs have emerged as a revolutionary technology with vast potential to transform various domains through their capability to monitor, collect, and process data from the physical world. This section provides an overview of the applications of WSNs and their significance in addressing contemporary challenges across diverse fields.

Environmental Monitoring

WSNs play a crucial role in numerous environmental surveillance applications, such as evaluating air and water quality, monitoring soil conditions, and preserving biodiversity. They allow for the persistent and detailed observation of environmental factors across various spatial and temporal dimensions. In the context of air quality monitoring, sensor nodes fitted with gas sensors can detect pollutants like carbon monoxide, nitrogen dioxide, and particulate matter in cities, industrial areas, and traffic-dense regions [56]. The strategic placement of WSNs allows environmental organizations and policymakers to access real-time air quality data, assisting in assessing pollution levels, pinpointing pollution sources, and executing effective countermeasures to protect public health and the environment [38].

Likewise, WSNs are widely used in water quality monitoring to evaluate a range of physicochemical and biological factors such as pH, dissolved oxygen, turbidity, conductivity, and nutrient concentrations in bodies of water [57]. Sensor nodes positioned in rivers, lakes, and coastal regions can identify alterations in water quality caused by natural phenomena, human activities, and environmental disruptions. The continuous gathering of data on water quality parameters made possible by WSNs allows for the early identification of water pollution incidents, algal blooms, and contamination events, facilitating prompt actions to safeguard aquatic ecosystems and potable water sources.

Industrial Automation

WSNs have revolutionized the field of industrial automation by offering real-time surveillance, control, and enhancement of manufacturing operations, machinery, and infrastructure. They are essential in manufacturing units and factories for monitoring the condition and performing predictive maintenance of vital machinery and equipment. By installing sensor nodes on important components such as motors, pumps, and turbines, WSNs facilitate uninterrupted monitoring of operational metrics like temperature, vibration, and pressure. This real-time surveillance allows maintenance staff to identify irregularities, predict potential breakdowns, and plan preventive maintenance, thereby reducing downtime, cutting down maintenance expenses, and prolonging the lifespan of equipment [58]. Moreover, WSNs are instrumental in tracking and managing assets within industrial settings by providing real-time location data and status updates of equipment, tools, and inventory items.

Healthcare

A key application of WSNs in healthcare is the ability to monitor patients remotely, providing continuous observation of patients' vital signs, physiological parameters, and activity levels beyond the confines of traditional clinical environments. Through the use of wearable sensors and mobile health devices equipped with WSN technology, patients can send real-time data on various health metrics such as heart rate, blood pressure, blood glucose levels, and physical activity to healthcare professionals. This enables remote monitoring and management of chronic diseases like diabetes, hypertension, and cardiovascular diseases. Solutions powered by WSN for remote monitoring encourage patients to take an active role in managing their health, leading to improved adherence to treatment plans and a decrease in hospital readmissions [59]. Furthermore, WSNs play a crucial role in elderly care and assisted living applications by keeping track of the health status, mobility, and safety of seniors in their homes and community settings. Sensor nodes incorporated into smart home devices, wearable gadgets, and environmental sensors can identify incidents like falls and wandering behavior among elderly individuals.

WSNs also have the potential to transform healthcare delivery by enabling telemedicine and telehealth services, reducing the need for in-person visits and making healthcare more accessible, especially in remote and underserved areas. Moreover, the data collected by WSNs can be used for predictive analytics, helping to identify health risks and intervene before the onset of serious health issues. This proactive approach to healthcare, enabled by WSNs, could lead to improved patient care and more efficient use of healthcare resources, transforming the healthcare landscape [39].

Smart Agriculture

WSNs have become a pivotal tool in the realm of smart agriculture, transforming traditional farming methods through precision agriculture, efficient resource management, and sustainable crop practices. In the context of precision farming, WSNs are strategically placed across agricultural fields to monitor soil conditions, climatic parameters, and crop health in real-time. By collecting data on factors such as soil moisture, temperature, pH levels, and nutrient concentrations, WSNs equip farmers with the necessary information to make educated decisions about irrigation timing, fertilization techniques, and pest control. Practices like variable rate irrigation and site-specific nutrient application, facilitated by WSNs, optimize the use of resources, reduce input costs, and increase crop yields, thereby boosting profitability and promoting environmental sustainability in farming operations [60].

Furthermore, WSNs play a crucial role in managing and conserving water within agriculture by monitoring water usage, detecting leaks, and improving irrigation strategies [60]. With sensor nodes installed in irrigation systems, soil moisture sensors, and weather stations, real-time data on soil moisture levels, weather predictions, and evapotranspiration rates are made available. This enables farmers to devise precise irrigation plans and minimize water wastage. The use of WSN data for smart irrigation scheduling and water distribution aids in preserving water resources, mitigating the risk of drought [40].

Military and Defense

WSNs have become essential tools in military and defense applications, offering real-time situational awareness, strategic intelligence, and battlefield surveillance capabilities. One of the main uses of WSNs in military operations is for battlefield surveillance and reconnaissance. In this context, sensor nodes are strategically placed to monitor enemy movements, identify threats, and gather real-time intelligence [61]. These sensor nodes, equipped with cameras, acoustic sensors, and motion sensors, can gather data on enemy troop movements, vehicle convoys, and weapon deployments. This allows commanders to evaluate the battlefield situation, pinpoint high-value targets, and plan tactical maneuvers effectively. By utilizing WSN data for intelligence, surveillance, and reconnaissance (ISR) operations, military forces can gain a competitive edge, maintain situational awareness, and achieve mission objectives with minimal risk to personnel and resources [61]. In addition, WSNs aid in perimeter security and force protection applications in military installations, forward operating bases, and critical infrastructure sites. Sensor nodes placed along perimeter fences, access points, and entryways can detect intrusions, breaches, and unauthorized access attempts

WSNs also facilitate asset tracking and logistics management in military supply chains, transportation networks, and deployment operations. Sensor nodes embedded in vehicles, containers, and cargo shipments can provide real-time data on asset location, condition, and status throughout the supply chain. This allows logistics planners to optimize transportation routes, allocate resources efficiently, and streamline deployment processes. By integrating WSN data with logistics management systems and command and control (C2) platforms, military organizations can enhance supply chain visibility, improve asset visibility, and enhance operational readiness and responsiveness in dynamic and complex environments [62].

2.2 Wireless Communication and Jamming Attack

Wireless communication has become an integral part of our modern world, revolutionized how we access information and interact with each other. One significant advancement stemming from wireless communication is the emergence of wireless sensor networks. These networks are comprised of spatially distributed autonomous sensors that communicate wirelessly, allowing for the monitoring of various physical or environmental conditions.

2.2.1 Overview of Wireless Communication

Wireless communication harnesses the power of electromagnetic waves to facilitate the transmission of information over varying distances without the need for physical connections. These waves, which are integral to wireless communication, are defined by characteristics such as frequency, wavelength, amplitude, and speed of propagation. They are generated by the fluctuation of electric and magnetic fields and traverse through space at the speed of light [63]. Among the various types of electromagnetic waves, radio waves are predominantly used in wireless communication. Their extensive use can be attributed to their unique ability to cover vast distances without the requirement of a physical medium.

The electromagnetic spectrum encompasses the entire range of frequencies of electromagnetic waves. It includes radio waves, microwaves, infrared, visible light, ultraviolet, X-rays, and gamma rays. Different frequency ranges within the spectrum are allocated for various wireless applications based on their specific characteristics. Particularly the radio frequency spectrum is used for wireless communication [58]. These waves carry data invisibly, enabling seamless communication between devices without the limitations of physical connections. Lower frequency bands are suitable for long-distance communication, while higher frequency bands support high data rates and shorter transmission distances.

Regulatory bodies allocate specific frequency bands for different wireless communication technologies to avoid interference and ensure efficient spectrum utilization [64]. The communication process involves encoding the data into the RF signal, modulating the signal for transmission, receiving the signal at the destination, and demodulating it to extract the original data.

In wireless communication, the transmission of information is achieved through the modulation of electromagnetic waves. Modulation is the process of varying one or more properties of a carrier signal, such as amplitude, frequency, or phase, in accordance with the information signal. Various modulation techniques are used to convert information into radio signals [63]. Amplitude Modulation (AM) involves varying the amplitude of the carrier wave to encode information. Frequency Modulation (FM) varies the frequency of the carrier wave, while Phase Modulation (PM) alters the phase of the carrier wave. Each modulation technique has its advantages and is used based on specific requirements such as signal fidelity, bandwidth efficiency, and noise immunity [64]. At the receiver, demodulation techniques are employed to recover the original information from the modulated carrier signal. Demodulation involves extracting the modulating signal from the modulated carrier wave. Different demodulation techniques are used depending on the modulation scheme employed in the transmission [64].

The wireless channel serves as the medium through which signals travel from the transmitter to the receiver. The wireless channel is affected by various propagation mechanisms that can impact signal strength and quality. Fading is a common phenomenon in wireless communication caused by changes in the wireless channel over time or distance. It can be categorized into path loss and multipath fading [65]. Path loss occurs due to the attenuation of the signal as it travels through the wireless medium. Multipath fading results from the superposition of multiple reflected signals reaching the receiver with different phases and amplitudes, causing constructive or destructive interference. Fading can significantly affect signal reception and must be considered in the design of wireless communication systems.

2.2.2 Overview of Jamming Attack

Radio communication is susceptible to interference from various sources, both unintentional and intentional [14]. Intentional interference, specifically jamming attacks, has received significant attention in the literature due to its disruptive nature. Jamming attacks represent a formidable threat to the integrity and reliability of WSNs, capable of disrupting communication channels and compromising the delivery of critical data. These attacks occur when adversaries deliberately interfere with wireless communication

signals, aiming to disrupt or block legitimate transmissions within the network. In essence, a jamming attacker floods the communication medium with noise or false signals, rendering it unusable for legitimate devices or nodes [8].

The common characteristic for all jamming attacks is that their communications do not adhere to MAC protocols. Jammers can disrupt legitimate wireless communication by either blocking the transmission of valid data or interfering with the reception of such data. Consider two legitimate wireless nodes, A and B, and a jammer, X. The jammer can prevent node A from transmitting packets by continuously transmit signals on the same channel, preventing A from detecting the channel as idle, a requirement for transmission. Alternatively, X can flood the channel with regular data packets, causing A to constantly receive non-essential or junk data. Even if node A successfully transmits a packet to node B, the jammer X can interfere during the reception phase. By broadcasting a strong radio signal, X can corrupt the message that B receives, degrading the communication reliability between the legitimate participants [2].

Modeling jamming attacks involves capturing the interference introduced by the jammer into the communication channel. One common approach is to model the jamming signal as an additive noise component that corrupts the received signal [5]. Mathematically, this can be represented as:

$$y[n] = x[n] + j[n] + w[n] \quad (1)$$

Where:

- $y[n]$ is the received signal corrupted by jamming and noise,
- $x[n]$ is the received signal corrupted by jamming and noise,
- $j[n]$ is the jamming signal introduced by the attacker, and
- $w[n]$ is the background noise present in the environment.

The effectiveness of a jamming attack can be quantified using metrics such as the Signal-to-Interference-plus-Noise Ratio (SINR), which compares the strength of the desired signal to the combined interference and noise level.

2.2.3 Types of Jamming Strategies

In the literature, jamming attacks strategies on WSNs are broadly classified based on their operational characteristics. This classification includes constant jamming, reactive jamming, and periodic jamming, each having unique propagation methods and varying effects on network performance [8, 16].

Constant jamming, a form of electronic warfare, involves the constant transmission of jamming signals across specific frequencies within the communication band. This attack method floods the communication channel with persistent interference, obstructing the receipt of legitimate data packets by the intended receivers. constant jamming attacks are relatively simple to execute and can disrupt multiple channels at once, making them a common threat in WSN settings [2]. Constant jamming attacks can drastically impair the performance of WSNs, resulting in communication disruptions and jeopardizing the accuracy of the gathered data. Moreover, constant jamming attacks can considerably affect the reliability and efficiency of WSNs in diverse applications, including environmental monitoring, surveillance, and healthcare [18]. Countering the effects of constant jamming attacks on WSNs is essential for guaranteeing the reliability and security of these networks. A variety of methods have been suggested to detect and counteract constant jamming attacks in WSNs. Adaptive frequency hopping techniques can be used to dynamically alter the operating frequency of WSN nodes, making it harder for the jammer to interrupt communication. In addition, energy-efficient jamming detection algorithms can be implemented to detect and counteract constant jamming attacks in real-time [1, 14]. These methods can boost the resilience of WSNs against constant jamming attacks, ensuring uninterrupted operation in demanding environments.

Reactive jamming is a dynamic strategy that adjusts jamming tactics based on observed network conditions and communication activities. Adversaries using reactive jamming selectively interfere with transmissions based on set criteria such as packet types, source addresses, or signal strength levels. By smartly reacting to network stimuli, reactive jammers aim to maximize disruption while minimizing detection, creating challenges for traditional jamming detection and mitigation mechanisms [8]. Reactive jamming attacks are especially damaging in WSNs, where nodes often have limited resources and lack advanced detection capabilities. Reactive jamming attacks can exploit weaknesses in WSN protocols, leading to significant disruptions in network communication and compromising the accuracy of collected data [7]. To combat the risk of reactive jamming attacks in WSNs, several defense mechanisms have been suggested by researchers. Anomaly detection techniques can be used to identify unusual

communication patterns indicative of reactive jamming activity [7, 13]. Additionally, cooperative jamming detection schemes, where nodes collaboratively analyze network traffic and share information about potential jamming attacks, can improve the resilience of WSNs against reactive jammers. These proactive defense mechanisms can assist in detecting and mitigating reactive jamming attacks in real-time, ensuring the dependable and secure operation of WSNs in adversarial environments [63].

Periodic jamming, also referred to as intermittent jamming, involves the periodic emission of brief bursts of jamming signals. Unlike constant jamming attacks that maintain a steady transmission, periodic jamming periodically disrupts communication channels, resulting in irregular interference. This type of attack can take advantage of weaknesses in communication protocols and synchronization mechanisms, leading to packet loss, synchronization errors, and transmission delays within the network. Periodic jamming attacks can have a significant impact on the performance of WSNs, breaking crucial communication links and jeopardizing the accuracy of the data collected [8]. To reduce the effects of periodic jamming attacks on WSNs, the creation of robust defense mechanisms is necessary. One method is to use intelligent jamming detection and avoidance techniques that can adapt to changing jamming patterns [2]. Furthermore, strategies such as channel hopping and frequency agility can be used to dynamically change communication channels in response to detected jamming activity. These proactive defense mechanisms can increase the resilience of WSNs against periodic jamming attacks, ensuring consistent and uninterrupted communication in the face of adversarial interference [64].

2.2.4 Impact of Jamming Attack on WSN

Jamming attacks in WSNs present significant challenges to the reliability, security, and functionality of these networks, especially in critical applications such as military surveillance, healthcare monitoring, industrial automation, and environmental sensing. The impact of jamming on WSNs can be far-reaching, affecting various aspects of network performance and posing risks to the integrity of transmitted data [62]. One of the primary consequences of jamming is the disruption of communication between sensor nodes, which hinders their ability to transmit data to a central base station. This disruption could leads to the loss of critical information, a particularly serious issue in time-sensitive applications like environmental monitoring or battlefield surveillance, where real-time data collection is essential for decision-making [8].

Another major impact of jamming attacks is the depletion of node resources, including battery power and processing capacity. Jamming techniques often induce packet loss, forcing sensor nodes into continuous retransmissions, which consumes additional energy and processing resources [12]. In WSNs, where nodes are typically powered by limited batteries and are difficult to recharge or replace, this rapid resource depletion can lead to premature node failure, thereby reducing the overall lifespan and reliability of the network [10]. Furthermore, the noise generated by jamming attacks can create security vulnerabilities by masking other malicious activities within the network. This distraction allows attackers to perform more sophisticated intrusions, such as injecting false data or intercepting legitimate communications, without being detected. In such cases, the integrity of the data within the network is compromised, which can have severe consequences in applications where data accuracy is paramount [22].

Jamming also poses a significant threat to time-sensitive applications by causing delays or disruptions in communication, which are particularly problematic in environments like healthcare and industrial control systems. Even brief communication failures can lead to critical issues, such as delayed patient monitoring alerts or operational failures in industrial automation systems, highlighting the vulnerability of these systems to jamming attacks [39, 58]. In addition, frequent communication failures due to jamming can undermine the reliability and trustworthiness of the network, especially in mission-critical scenarios. As jamming-induced disruptions accumulate, confidence in the network's ability to deliver timely and accurate data diminishes, which can ultimately result in the abandonment of the network or its redesign.

Furthermore, jamming attacks can selectively target and isolate individual nodes from the network, effectively preventing them from transmitting data. This form of targeted disruption is particularly harmful in WSNs designed to provide comprehensive coverage, as it limits the network's ability to monitor certain areas or regions. This selective isolation can facilitate undetected malicious activities in specific locations, compromising the overall security of the system [6]. Lastly, the instability induced by jamming attacks can trigger false alarms, leading to unnecessary responses or confusion within the monitoring system. This not only hampers the efficiency of network management but also complicates decision-making processes, further reducing the operational effectiveness of the network.

In conclusion, jamming attacks in WSNs have wide-ranging impacts, including communication disruptions, resource depletion, compromised data integrity, and network instability. These attacks threaten the overall functionality and security of the network, particularly in critical applications where

reliable, real-time data is essential [8]. As the prevalence of WSNs in mission-critical environments increases, addressing the challenges posed by jamming attacks becomes imperative to ensure the continued performance and reliability of these networks.

2.2.5 Metrics for Detection of Jamming Attack

Over the last decade, detecting jamming attacks in WSNs has been a prominent area of research. Existing methods often depend on specialized devices or algorithms embedded within sensor nodes. These approaches leverage a priori information about communication behavior under normal and jammed conditions, which can be monitored using indicators and metrics from various network layers [5, 8, 16].

Packet Delivery Ratio (PDR) is a key measure used in WSNs to assess the reliability and efficiency of data transmission. It quantifies the ratio of packets successfully acknowledged by the receiving node to the total packets sent by the transmitting node during a communication session. PDR is particularly important for assessing network performance in settings susceptible to interference and jamming attacks [10]. The calculation of PDR involves the exchange of packets between sender and receiver nodes, ending with the transmission of acknowledgment (ACK) packets when data is successfully received. This is typically enabled through a 4-way handshaking mechanism (RTS/CTS/Data/Ack), which allows nodes to verify successful data delivery. The PDR measurements can differ based on the network protocol in use. In reliable protocols like the Transmission Control Protocol (TCP), where acknowledgments are produced for each successfully delivered packet, PDR can be monitored at both the sending and receiving ends [2]. However, using PDR as a metric for detecting jamming may have limitations in sensor nodes with resource constraints operating under TCP protocols due to the overhead of acknowledgments [8].

The Packet Sending Ratio (PSR) is a vital measure used to measure the efficacy of data transmission and assess the influence of jamming attacks on network functionality. PSR quantifies the proportion of packets that a node successfully transmits within a given timeframe to the total packets it intends to transmit in that duration. The computation of PSR involves determining the availability of the communication channel for the transmitting node and multiplying it by the transmission rate. This computation provides an indication into the efficiency of packet delivery and the degree of network congestion or interference [8]. The value of PSR becomes apparent in the detection and quantification of jamming attacks. Malicious nodes can interrupt communication channels, leading to packet loss and a decline in network performance. By tracking changes in PSR, unusual patterns suggesting jamming

activities can be identified, enabling the implementation of suitable measures to mitigate their effects. However, it is essential to consider network dynamics and environmental factors when interpreting PSR values [16]. PSR variations can result from legitimate network congestion or environmental disruptions. Thus, supplementary contextual data is required to distinguish between normal and abnormal conditions.

Bit Error Rate (BER) is a critical metric utilized in wireless networks to quantify the accuracy of data transmission by measuring the ratio of corrupted bits to the total bits received during a communication session [1]. BER serves as a key indicator of channel quality and network reliability, offering insights into the presence of transmission errors and potential vulnerabilities to external interference, including jamming attacks. BER computation typically involves comparing received data bits against their expected values and calculating the proportion of discrepancies observed. Higher BER values indicate a greater likelihood of transmission errors, which can impede data accuracy and compromise the overall effectiveness of the network [3]. While BER is effective in detecting reactive jamming attacks, its computation can be computationally intensive, particularly in scenarios where nodes must monitor BER for multiple radio links within the network [2]

Energy Consumption Amount (ECA) serves as a pivotal metric in WSNs for evaluating the energy usage of individual sensor nodes over a specified time period. It quantifies the approximate amount of energy consumed by a node, providing insights into network longevity, resource utilization, and susceptibility to energy-related attacks, including jamming. ECA computation typically involves measuring the voltage drop across the battery component of a sensor node and estimating energy consumption based on factors such as load characteristics, transmission rates, and operating conditions [4]. One common challenge associated with ECA measurement is the difficulty in accurately sampling energy consumption under varying traffic loads and network conditions. In dynamic environments where traffic patterns fluctuate unpredictably, traditional energy estimation techniques may yield imprecise results, limiting the effectiveness of energy-aware protocols and algorithms [3]. Moreover, detecting jamming attacks using ECA metrics poses additional challenges, particularly when adversaries employ sophisticated power manipulation techniques to evade detection [4].

Received Signal Strength Indicator (RSSI) is a crucial metric utilized in WSNs to quantify the strength of radio frequency signals received by sensor nodes from neighboring devices or base stations. RSSI serves as a fundamental indicator of communication quality, providing insights into signal propagation, link reliability, and network performance. RSSI measurement typically involves capturing and

analyzing radio frequency signals received by sensor nodes using dedicated hardware or integrated radio transceivers [10]. One common challenge associated with RSSI measurement is the influence of environmental factors and interference sources on signal quality. Variations in signal propagation, multipath fading, and electromagnetic interference can affect RSSI accuracy and reliability, leading to erroneous signal strength estimations [3]. Moreover, interpreting RSSI values requires careful consideration of distance, obstacles, and radio frequency interference patterns within the network environment. In densely populated or urban areas, signal attenuation effects and multipath propagation phenomena may distort RSSI readings, complicating signal strength estimation [5].

Packet Error Rate (PER) is a key metric in wireless communication systems, including WSNs, used to assess the reliability of data transmission. It is defined as the ratio of error packets, even those with a single erroneous bit after Forward Error Correction (FEC), to the total number of received packets [13]. PER offers critical insights into the robustness of a communication link. Compared to Bit Error Rate (BER), PER is more suitable for jamming scenarios because detecting a jammer is unnecessary if it does not significantly disrupt the transmission or reception of entire packets [13].

2.3 Exponentially Weighted Moving Average

2.3.1 Overview of EWMA

EWMA, which was proposed by Roberts [66], is an efficient statistical technique used in detecting small shifts in time-series data. It functions by first defining a threshold that delimits a standard behavior before periodically handling updates on average of the observed data traffic. It is a widely used statistical method for analyzing time-series data and detecting anomalies. In the context of WSNs, EWMA has been extensively applied for detecting abnormalities in sensor readings and anomaly detection, such as those caused by environmental disturbances [67].

The EWMA algorithm calculates a weighted average of historical data points, assigning exponentially decreasing weights to older observations. Mathematically, the EWMA value $S(t)$ at time t is computed as a combination of the current observation $X(t)$ and the previous EWMA value $S(t-1)$ using the following recursive formula shown in Equation (2) [68]:

$$S(t) = \alpha X(t) + (1-\alpha).\ S(t-1) \quad (2)$$

Where:

- S(t) is the current EWMA value,
- X(t) is the current observation at time t,
- S(t−1) is the previous EWMA value,
- α is the smoothing constant, also known as the weighting parameter.

The smoothing constant α, which ranges between 0 and 1, significantly influences the behavior of the EWMA. While scholars have proposed various α values, the choice depends on both the size of the mean shift and the in-control Average Run Length [5].

When α is close to 1, the EWMA becomes highly sensitive to changes in the observations. This means that the current observation X(t) heavily influences the EWMA value, making it responsive to short-term fluctuations [70]. This high sensitivity is beneficial in scenarios where capturing rapid changes is crucial. However, it also makes the EWMA more susceptible to noise and short-term variability [67].

Conversely, when α is close to 0, the EWMA places more weight on the historical average S(t−1) and less on the current observation X(t), resulting in greater smoothing. This makes the EWMA less responsive to short-term fluctuations, providing a more stable and noise-resistant measure [69]. Such a setting is advantageous for applications focused on long-term trends. However, a low α can delay the detection of recent changes or trends [67]. The appropriate choice of α depends on the specific requirements of the application. For high-frequency data that necessitates quick detection of changes, a higher α is preferable [70]. For applications prioritizing noise reduction and long-term trend analysis, a lower α is more suitable. Often, the optimal α value is determined through a combination of domain knowledge and empirical testing [67].

Typically, α is selected to be less than 0.5. However, choosing a very small α may reduce the algorithm's sensitivity to attacks with moderate intensity or short duration. Thus, in practical applications, α typically falls between 0.2 and 0.5 [5].

The estimated variance σ_z^2 of the EWMA statistic can be approximated using [68]:

$$\sigma_z^2 = \left(\frac{\alpha}{2-\alpha}\right)\sigma_x^2 \tag{3}$$

Where σ_x is the standard deviation obtained from the historical data and α is the smoothing factor.

The control chart center line (CL) is the target value, which is the mean of the data points; therefore, the upper control limit (UCL) and lower control limits (LCL) can be determined using the Equation (4) [68]:

$$\begin{aligned} LCL &= CL - f * \sigma_z \\ UCL &= CL + f * \sigma_z \end{aligned} \tag{4}$$

Where the factor f is set to be equal to 3-sigma control limits and σ_z is the variance of EWMA.

In WSNs, EWMA is commonly employed for detecting abnormal sensor readings indicative of sensor malfunctions. By continuously monitoring sensor data streams and computing EWMA values, anomalies such as sudden spikes or persistent deviations from the expected trend can be identified, triggering alarm notifications for adaptive response [69]. Moreover, EWMA can be augmented with machine learning techniques to improve anomaly detection accuracy to evolving attack patterns [71]. In the context of jamming detection in WSNs, the smoothed value calculated by EWMA is compared against the predefined UCL and LCL. If the smoothed value falls outside these control limits, it is flagged as an anomaly, indicating a potential jamming attack. This approach effectively identifies deviations from normal network behavior, enabling timely detection of jamming activities [5].

2.3.2 Advantages and Limitations of EWMA

EWMA is a statistical tool widely used in time series analysis due to its numerous advantages, yet it also comes with inherent limitations. This section provides the advantages and limitations of EWMA, drawing insights from existing literature.

One of the significant advantages of EWMA is its adaptive sensitivity to changes in data distributions over time. This adaptive nature allows EWMA to capture underlying trends and changes in the data stream effectively. Mahmood *et al.* [69] demonstrates how EWMA adjusts its weighting scheme to give

more weight to recent observations, making it responsive to sudden changes or anomalies in time series data. This adaptability enables EWMA to detect trends and anomalies in real-time, making it a valuable tool for monitoring dynamic systems and processes.

Another advantage of EWMA is its robustness to noise in time series data [68]. By assigning exponentially decreasing weights to older observations, EWMA smooths out short-term fluctuations and random variations, focusing on the underlying trend of the data stream. This capability is particularly beneficial for machine learning, as it ensures that input features are less affected by noise, leading to improved model performance and more accurate predictions. Viinikka and Debar [70] highlight how EWMA effectively filters out noise and outliers, providing a clearer representation of the underlying data dynamics. This robustness makes EWMA suitable for analyzing noisy data streams common in various domains. By reducing noise and outliers, data smoothing enhances the quality of input features, thereby improving the performance and accuracy of machine learning classifiers.

Despite its advantages, EWMA also has limitations that warrant consideration. One such limitation is its susceptibility to trend shifts or sudden changes in data distributions. While EWMA is designed to adapt to changes over time, it may exhibit delayed responses to significant shifts in data trends [67]. Moreover, an abrupt change in the underlying data dynamics can distort EWMA estimates, leading to inaccurate predictions or false alarms [71]. This limitation underscores the importance of carefully monitoring and validating EWMA outputs in dynamic environments. Moreover, EWMA's sensitivity to outliers in time series data can pose challenges in certain applications. While EWMA effectively filters out noise and random variations, outliers with disproportionately large or small values may exert a significant influence on the moving average, skewing its estimation of the underlying trend [69]. This sensitivity to outliers necessitates careful preprocessing and outlier detection techniques to ensure the integrity and reliability of EWMA-based analysis [68].

2.4 Summary

In this Chapter we have provided a comprehensive review of the existing literature on WSNs and jamming attack. It begins by discussing the architecture and characteristics of WSNs, highlighting their application in various fields such as environmental monitoring, healthcare, and industrial automation. The chapter then delves into the concept of wireless communication and different types of jamming attacks, including constant, reactive, and periodic jamming. The impact of these attacks on the

performance of WSNs is explored, emphasizing the necessity for effective detection and classification mechanisms to maintain network reliability. Further, various detection metrics for jamming attack detection discussed in the literature were reviewed, offering insights for selecting suitable metrics for jamming attack detection in WSNs. The EWMA method is also introduced as a technique for detecting anomalies by smoothing time-series data, although its limitation due to manual tuning of threshold is noted. Furthermore, we explore how EWMA can enhance machine learning accuracy by serving as a data smoothing technique to process input features.

Chapter Three: Related Work

3.1 Introduction

WSNs have become an integral part of various applications, including environmental monitoring, healthcare, industrial automation, and military surveillance, owing to their ability to collect data from remote or hostile environments. However, the open nature of the wireless medium makes WSNs vulnerable to various security threats, with jamming attacks being one of the most serious and prevalent [1]. Jamming attacks involve the deliberate interference with wireless communication within a network by transmitting radio frequency signals. These attacks can disrupt the normal functioning of WSNs, leading to the loss of critical data, degradation of network performance, and even denial of service. Therefore, the detection and classification of jamming attacks have emerged as crucial research areas in ensuring the security and reliability of WSNs [8].

In this chapter, we review existing literature on jamming attack detection and classification in WSNs. We categorize the existing approaches into two main categories: statistical-based methods and machine learning-based methods. Statistical methods rely on analyzing various statistical features of the received signals to detect the presence of jamming attacks. Machine learning-based methods utilize supervised, unsupervised, or semi-supervised learning algorithms to classify normal and jammed traffic patterns. By examining the strengths and limitations of each approach, we aim to provide insights into the state-of-the-art techniques for jamming attack detection and classification in WSNs. Finally, the common gaps of the reviewed works as well as the way the newly proposed algorithm fills those gaps is described briefly.

3.2 Statistical-Based Methods

Statistical methods for jamming attack detection rely on analyzing various statistical features of the received signals to differentiate between normal network behavior and the presence of jamming attacks [8, 16]. These methods are based on the premise that jamming attacks introduce anomalous patterns in wireless communication that can be detected through statistical analysis. By monitoring and analyzing metrics such as signal strength, traffic patterns, and packet characteristics, statistical methods aim to identify deviations from expected behavior caused by jamming attacks [5]. In this section, we provide

an in-depth review of the statistical-based jamming attack detection methods proposed in the literature for jamming attack detection in WSNs.

Muhammed *et al*. [2] proposed a three edge nodes scheme, which have different transmission frequencies in the same bandwidth. By having each edge node operate on a different transmission frequency, the scheme ensures that even if one channel is jammed, the network remains operational. A key aspect of the edge node scheme is its use of Round-Trip Time (RTT) of transmitting signals. Before transmitting a packet to the transmission media, all three edge nodes calculate the estimated RTT of the packet. This estimated RTT value is then included in the payload of the packet. At the remote location, this value is compared to the calculated estimator value of RTT to ensure the maintenance of transmission reliability. The RTT is used to detect disturbances in transmission, which serve as an indication of a potential jamming attack. If an attacker jams one of the transmission channels, the other two edge nodes check the serviceability of the medium by transmitting information from the same deployed WSNs. Additionally, the RTT of the adjacent channel deviates from its specified interval due to high-frequency interference in the neighboring channels, signaling a jamming attack in the network. The authors achieved a jamming attack detection rate of around 94% in their experiments conducted using the OMNeT++ simulator, which is significantly higher than existing schemes. The use of RTT and frequency variation enhances the reliability of jamming attack detection. Additionally, the approach offers a simple operational deployment and does not require special hardware or software. However, the approach has certain limitations. It may not be as effective in large or inaccessible regional-based WSNs due to the difficulty in monitoring the edge nodes. The model's effectiveness can also be compromised by high interference frequencies in adjacent channels, which can disrupt the consistency of RTT measurements. Furthermore, the approach assumes that an attacker cannot jam all channels simultaneously, which might not hold true in all scenarios, potentially leading to undetected jamming attacks.

Osanaiye *et al*. [5] proposed a step-wise approach using a statistical process control technique to detect jamming attacks in WSN. The authors deploy EWMA, a statistical process control technique. The method involves monitoring packet inter-arrival times to detect potential anomalies indicative of a jamming attack. The EWMA model is a type of moving average model that gives more weight to recent observations, making it highly sensitive to recent changes in the data. This characteristic makes it particularly suitable for detecting anomalies in packet inter-arrival times, which could indicate a

jamming attack. The detection process begins by collecting packet inter-arrival times from the network traffic. These times are then input into the EWMA model, which calculates the moving average, giving more weight to recent observations. If the calculated EWMA value exceeds a predefined threshold, it is considered an indication of a potential jamming attack. This approach's main advantage is its simplicity and efficiency. By focusing on a single metric, the packet inter-arrival time, the detection process is significantly simplified, reducing the computational overhead. The paper also presents the results obtained from trace-driven simulations to validate the proposed method. The results show that the proposed method can effectively detect jamming attacks with 100% accuracy for scenarios involving 20 or more jammed packets. Despite these promising results, the paper does have some limitations. The proposed solution relies heavily on the assumption that packet inter-arrival times are the most effective metric for detecting jamming attacks. While this approach simplifies the detection process, it may overlook other potential attack indicators, leading to a higher false positive rate, which is not clearly specified in their paper. Furthermore, the effectiveness of the proposed solution is demonstrated through simulation, which, although valuable, may not fully capture the complexities and unpredictability of real-world scenarios.

3.3 Machine Learning-Based Methods

Feng *et al.* [10] presents a machine learning-based approach for detecting and classifying jamming attacks in wireless networks. The authors implemented jamming attack modules using the NS-3 simulator to study different jamming strategies, including constant, random, and reactive jammers. The proposed detection schemes rely on several machine learning algorithms, such as K-Nearest Neighbors (KNN), Decision Trees, and Random Forests. Key metrics used for detection include received signal strength (RSS), carrier sense time, noise, and packet delivery rate (PDR). The machine learning models are trained and evaluated using simulation data, with 70% of the data used for training and 30% for testing. The machine learning-based detection methods demonstrated high accuracy in identifying and classifying jamming attacks. The Random Forest algorithm achieved the highest accuracy at 81%, followed by K-nearest neighbors with 79%, and decision trees with 75%. The study found that the Random Forest algorithm performed best in distinguishing between different jamming strategies, although the classification rates for reactive and random jammers were lower compared to constant jammers. The use of multiple metrics, such as RSS and PDR, improved the detection accuracy

compared to using a single metric. One significant limitation is the reduced accuracy indicating a need for further optimization of the detection algorithms.

Jacovic *et al.* [13] introduces an experimental platform for RF jamming detection and mitigation using software-defined radio (SDR) systems. The platform includes online jammer classification and utilizes reconfigurable beam-steering antennas at the physical layer. The classification component leverages machine learning, specifically Random Forest classifiers, trained with data collected over-the-air. The features used for classification include Received Signal Strength Indicator (RSSI), Error Vector Magnitude (EVM), and Packet Error Rate (PER). The classifiers detect the presence and type of jamming attack, which then triggers the mitigation mechanism using reconfigurable antennas to minimize the jamming effects. The experimental results demonstrated the effectiveness of the jamming mitigation system in both over-the-air and ray-tracing emulated environments. The Random Forest classifiers achieved high accuracy in detecting and classifying various types of jammers, such as constant, periodic, and reactive jammers. The integration of machine learning-based classification with reconfigurable antennas significantly improved the system's ability to maintain communication performance under jamming conditions. Specifically, the classifiers achieved an accuracy of 93% in multi-class classification and 95% in binary classification scenarios. However, among the features used, EVM has high computational overhead and may not be applicable for resource-constrained devices.

Kasturi *et al*. [14] introduces a machine learning-based approach for detecting and classifying jamming attacks in wireless networks. Using the ns-3 simulator, the study simulates three types of jamming attacks: constant, reactive, and random jammers. Network metrics such as Packet Delivery Ratio (PDR) and Received Signal Strength (RSS) are collected from the application and physical layers of the network. The Gradient Boosted Decision Tree Algorithm (GBDT) is applied to classify the types of jamming attacks, and the results are compared with other machine learning algorithms like Random Forest, Multi-layer Perceptron (MLP), K-Nearest Neighbors (KNN), and Decision Tree. The Gradient Boosting Algorithm outperformed other algorithms in terms of classification accuracy. The study reported a classification accuracy of 94.9% for the GBT, surpassing the Random Forest algorithm, which achieved 92.6%. The results showed that distinguishing between jamming and non-jamming scenarios was relatively easy, but classifying different types of jammers proved more challenging due to overlapping metrics like PDR and RSS. However, GBT successfully improved classification

accuracy over the other models. One limitation of the proposed approach is that boosting techniques, including GBT, have high computational overhead and are susceptible to overfitting, especially when the dataset is small.

Arjoune *et al.* [15] proposed a novel jamming attack detection approach based on machine learning for wireless communication. The author's methodology for detecting jamming attacks involved a comprehensive approach, which included the selection of relevant input features, precise data measurement and collection, and building an extensive dataset. This dataset plays a pivotal role in training, validating, and testing the proposed model. The authors relied on four key parameters: packet delivery ratio, bad packet ratio, received signal strength, and clear channel assessment to detect jamming attacks. To address the issue of underfitting, the paper incorporated key techniques such as dataset randomization and normalization, as well as the application of cross-validation methods. The authors utilized a variety of machine learning techniques, including Random Forest, support vector machine, and neural network. These techniques are trained, evaluated, and tested using a large dataset generated using signal features that identify jamming signals. The performance of these algorithms is evaluated and compared using several metrics, including the probability of detection, probability of false alarm, probability of miss detection, and accuracy. The results revealed that the Random Forest is effective achieving a 96.6% accuracy in detecting jamming attack. However, the dataset is gathered from a simulator and may not accurately represent real-world scenarios.

Testi *et al.* [12] introduces a novel machine learning based algorithm for jamming detection and classification in wireless networks, particularly aimed at Internet of Things (IoT) environments. The algorithm is designed to be implemented in the network gateway and employs an energy detector (ED) to identify jamming activities. Key features are extracted from the detected signals, followed by dimensionality reduction to streamline the data processing. The final step involves multi-class classification to distinguish between different types of jamming attacks. The method ensures rapid detection and classification while maintaining the privacy of network users by not requiring inspection of decoded information. The proposed single-hidden-layer feed-forward Neural Network algorithm demonstrates high accuracy in detecting and classifying jamming attacks, achieving up to 99% accuracy in some scenarios. Extensive numerical evaluations were conducted, considering various parameters such as the number of principal components selected through dimensionality reduction, observation window length, shadowing intensity, and signal-to-jammer ratio (SJR). The results indicate

that the algorithm outperforms existing state-of-the-art solutions, offering a highly reliable and efficient approach to jamming detection and classification in IoT networks. One major limitation is the potential impact of varying network conditions on the performance of the algorithm. Factors such as extreme shadowing or very low signal-to-jammer ratios may affect detection accuracy.

Greco *et al.* [16] presents a framework for jamming detection in drone networks utilizing a distributed machine learning approach. The proposed system uses supervised learning techniques such as MLP and decision trees to detect jamming attacks. The system operates by collecting network metrics, including throughput, packet delivery ratio, and received signal strength indicator which change during a jamming attack. These metrics are used to train the classifiers. The system evaluates publicly available datasets and simulates drone network datasets using NS-3. The performance of the classifiers was examined in different communication environments and topologies, with varying window sizes and overlapping time windows. The study shows that the MLP classifier performs better than the decision tree in larger and more complex communication scenarios, achieving a detection accuracy of 96%. The decision tree classifier showed an accuracy of about 50%. Both classifiers performed better when shorter sampling windows were used, and datasets with overlapping time windows resulted in higher classification accuracy. MLP proved more generalizable when applied to communication scenarios it wasn't specifically trained for. While the framework achieved high detection accuracy, there were limitations. The MLP classifier have higher computational overhead compared to decision tree, which may not be ideal for resource-constrained environments. Additionally, the study used datasets based on simulations and the real-world applicability of the framework was not thoroughly validated.

Upadhyaya *et al.* [11] proposed a machine learning-based approach for detecting jamming attacks in wireless IoT networks. The system is designed to be non-node-centric, reducing the overhead on IoT nodes by employing dedicated anchor nodes to passively collect network information. The algorithm initially focuses on using the radio signal strength indicator for jamming detection. Various machine learning techniques, such as Support Vector Machines, Decision Trees, Random Forests, and Real AdaBoost, are employed to classify jamming events. The authors also extend their approach by incorporating multi-path profile data, which improves detection accuracy. The system was evaluated using both simulated and real network data. In simulations, the RSSI-based detection method achieved an accuracy of 98.6%, while the real network data yielded 89.7% accuracy. By utilizing multi-path

profile information, the accuracy of the system improved further, reaching 99.01% in simulations. Random forest boosted with Real AdaBoost showed the best performance among the algorithms tested, with low false positive and false negative rates. The limitation is that while the algorithm performs well with simulated data, its accuracy decreases when applied to real-world network data, highlighting challenges in practical deployment.

3.4 Summary

The body of existing literature demonstrates that statistical methods have proven to be efficient in detecting jamming attacks within WSNs. The statistical method is well suited for resource-constrained sensor nodes as it imposes minimal overhead on the network. One of the limitations of statistical methods like EWMA lies in their primary design for anomaly detection rather than multi-class classification. However, EWMA can still be leveraged as a data smoothing technique, whose outputs can be integrated into machine learning models, such as Random Forest classifiers, to enable more effective multi-class classification. Differentiating between different types of jamming attacks allows for the implementation of appropriate countermeasures [8]. Additionally, statistical methods are prone to a high rate of false positives due to the challenge in differentiating between regular variations and genuine anomalies when relying on manually tuned threshold-based detection [6, 12].

On the other hand, machine learning-based jamming attack detection methods exhibit promising potential in accurately classifying jamming attacks. However, most related works focus on general wireless communication and fail to consider the resource constraints of sensor nodes. Implementing such techniques can introduce significant computational overhead and energy consumption, which is a critical concern in WSN environments [8]. However, the emergence of TinyML, a subset of machine learning that allows models to run on smaller and less powerful devices, allows the implementation of machine learning model on sensor nodes [77, 78]. Even though TinyML allows sensor node to run machine learning models locally the computational overhead is still higher than techniques such as statistical methods [75].

This research aims to overcome these limitations by leveraging the synergy between statistical-based and machine learning-based methods using EWMA and Random Forest. This approach is intended to address the limitations of each method individually while enhancing accuracy, reducing false positives, and improving the overall security and reliability of WSNs against evolving jamming attacks.

Chapter Four: Design of Jamming Attack Detection and Classification Method

4.1 Introduction

This chapter presents the proposed solution for detecting and classifying jamming attacks in WSNs using EWMA and a Random Forest Classifier. The chapter starts with a discussion of the design considerations for the proposed solution. This includes the choice of features for detecting jamming attacks, parameter optimization for EWMA, and the Random Forest classifier. Following this, the system architecture components of the proposed solution, along with a detailed description and complete algorithms, are presented, which include the data collection and preprocessing methods, the jamming detection algorithm, and the training of the Random Forest classifier model.

4.2 Design Considerations

During the design phase of the proposed solution for jamming attack detection and classification in WSNs, several key factors were carefully considered to ensure its effectiveness. The first factor is the sensor network architecture, where a cluster-based network design was assumed. In this architecture, sensor nodes are grouped into clusters, each managed by a cluster head. The cluster heads are responsible for detecting jamming attacks within their respective member nodes, while the base stations handle the detection of attacks targeting the cluster heads.

Another important factor is the jamming attack scenarios, as jamming attacks in WSNs can take various forms, each with unique characteristics and implications for detection and classification. Specifically, these scenarios include constant jamming, which continuously disrupts communication, periodic jamming, where disruptions occur at regular intervals, and reactive jamming, which activates only when it detects network activity.

Accuracy is also a crucial consideration in the design of the solution. To improve accuracy, different values for the weighting parameter in the Exponentially Weighted Moving Average (EWMA) were examined, and parameter optimization for the Random Forest Classifier was performed. The Random Forest classifier was chosen for its high accuracy in detecting and classifying jamming attacks. During

the evaluation phase, metrics such as accuracy, precision, recall, and F1-score were used to assess the performance of the solution.

Furthermore, minimizing overhead is essential for ensuring the efficiency of the solution. The selection of features for jamming attack detection and classification plays a critical role in reducing overhead. The Received Signal Strength Indicator (RSSI) and Packet Error Rate (PER) features were chosen based on their relevance to jamming attack detection, their availability in the dataset, and the associated overhead. Using RSSI, a readily available metric at sensor nodes, eliminates the need for additional hardware or algorithms, significantly reducing overhead. Meanwhile, the PER is calculated at the cluster head and base station, minimizing the amount of data transmitted and conserving both bandwidth and energy resources.

4.3 System Architecture

In this section, we present the architecture of the jamming attack detection and classification using EWMA and Random Forest Classifier. As shown in Figure 4.1 the proposed solution consists of two phases: Training phase and Testing phase.

During the training phase, the training dataset is processed by the EWMA (Exponentially Weighted Moving Average) processor, which consists of two components: the EWMA computation and the threshold calculation. The EWMA computation component computes the EWMA values for each feature in the training dataset. These newly computed values are then used for threshold calculation and for training the Random Forest classifier. The threshold calculation determines a threshold value for jamming detection by computing UCL (Upper Control Limit) and LCL (Lower Control Limit), which is subsequently passed to the EWMA-Based jamming detection component. The model training component trains the Random Forest classifier model using the output of EWMA computation of the training dataset, preparing it for jamming attack classification.

In the testing phase, the features from the testing dataset goes through EWMA computation, which are then used for jamming attack detection and classification these include RSSI (Received Signal Strength Indicator) and PER (Packet Error Rate). The EWMA computation smooths these input features by computing their EWMA values. These values are passed to the EWMA-Based jamming detection component. Using the threshold values obtained from the threshold calculation, the component determines if a jamming attack is occurring. Specifically, it compares the EWMA values of each feature

to the LCL and UCL computed for each feature by the threshold calculation. If the EWMA values fall within the LCL and UCL bounds, the network is considered normal; otherwise, a potential jamming attack is detected.

The jamming classification process is initiated only when the EWMA-Based jamming detection component determines that a potential jamming attack is occurring. If no jamming is detected, normal operations continue. Once a jamming attack is detected, the jamming classification component identifies the type of jamming attack being carried out. In addition to classifying jamming attacks, the Random Forest classifier also helps filter out false positive generated by the EWMA-based jamming detection method.

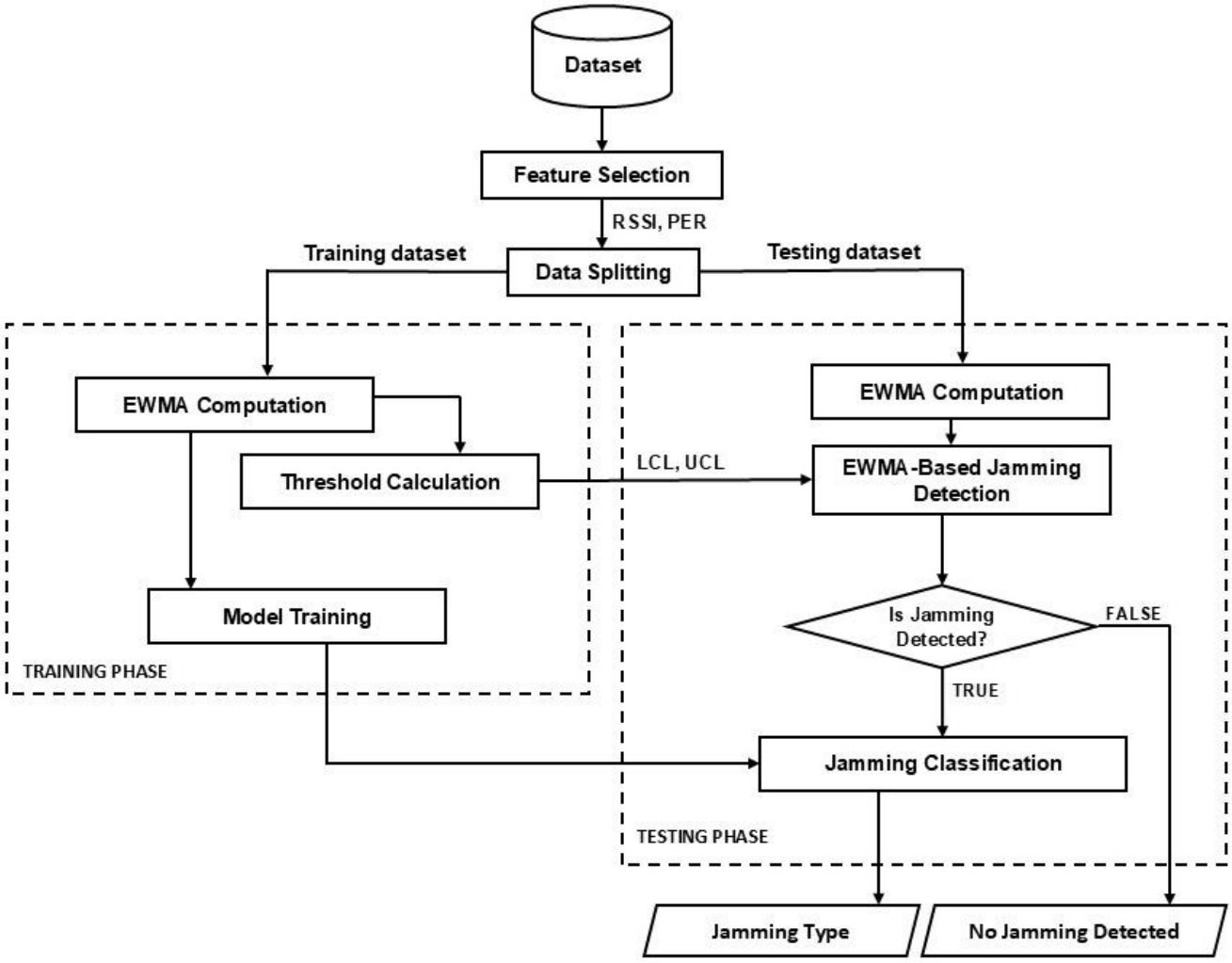


Figure 4.1: Architecture of the proposed Jamming Detection and Classification method

A detailed description of each component of the architecture of the proposed solution is provided below. It begins with the dataset utilized, feature selection and data splitting. The main component of the architecture is the training phase and testing phase.

Dataset

In most of the aforementioned studies in related work, analysis is limited to simulation and would benefit from experimental evaluation to provide a greater sense of realism to the communication devices considered. In this study, we utilized a dataset acquired from experimental evaluation conducted by Jacovic *et al.* [13]. During the training data collection experiments, the authors recorded logs of the features for each jamming case into separate files. Additionally, three unique cases were considered as normal class: normal operation, network congested links, and weak link conditions.

Feature Selection

The dataset initially included three features: Packet Error Rate (PER), Received Signal Strength Indicator (RSSI), and Error Vector Magnitude (EVM). RSSI is the measure of the power level received by a wireless device from a radio signal, indicating signal strength. PER is the ratio of incorrectly received packets to the total number of transmitted packets. When cluster heads or the base station receive a packet, the transceiver measures the RSSI of the incoming signal. The transceiver hardware automatically records the RSSI value during packet reception, so it doesn't pose an overhead. The number of erroneous packets after error correction is divided by the total number of received packets over a specific period to calculate the PER. However, EVM is excluded as it requires calculating the error between transmitted and received symbols in the constellation space. This involves per-symbol error calculations followed by root-mean-square computations, making it computationally intensive compared to RSSI and PER.

Data Splitting

During this stage, the dataset is divided into two sets: a training set and a testing set, with a ratio of 70% to 30%, respectively. The training set, consisting of 70% of the dataset, was used to train the Random Forest classifier and to calculate threshold values for LCL and UCL for EWMA-based jamming detection. Meanwhile, the testing set, comprising the remaining 30% of the dataset, was used to evaluate the performance of the trained model. This approach ensured that the model was trained on a diverse

range of jamming scenarios and tested on unseen data, thereby enhancing its ability to accurately classify jamming attack types.

4.3.1 Training Phase

The first step in designing the jamming attack detection and classification is training the model. This phase includes EWMA processing, in which EWMA computation and threshold calculation carried out, and model training. Below, we will discuss each component of the training phase in detail.

EWMA Computation

The EWMA computation component is a crucial part of the proposed solution, tasked with generating new features based on EWMA from the existing features. The EWMA computation component computes the EWMA value of the selected features using the Equation (5).

$$S(t)=\alpha X(t)+(1-\alpha).\ S(t-1) \qquad (5)$$

Where S(t) is the mean of the historical data, X(t) is the current observation at time t, S(t−1) is the previous EWMA value, and α is the smoothing constant, also known as the weighting parameter. The parameter α is crucial in determining how much older data influences EWMA statistics. When α is set to higher value, only recent measurements affect EWMA, giving them more weight, while older observations are given less weight. Higher values of α, like 0.9, prioritize recent observations, reducing the influence of older ones. Conversely, lower α values give more weight to older data. EWMA possesses a smoothing capability, effectively removing noise in time series data. The algorithm for EWMA computation is outlined in Algorithm 4.1.

Algorithm 4.1: EWMA Computation

```
Input: current_value, previous_ewma_value, alpha
Output: ewma_value
Begin
    If previous_ewma_value is not set
        ewma_value = current_value
        previous_ewma_val = current_value
        Return ewma_value
    Else
        ewma_value = alpha*current_value + (1-alpha) * previous_ewma_value
        previous_ewma_value = ewma_value
        Return ewma_value
    End If
End
```

This algorithm computes EWMA using three input parameters: alpha, the smoothing constant; current_value, representing the current feature value; and previous_ewma_value, which holds the previous EWMA value. If previous_ewma_value is not initialized, indicating the first calculation, the algorithm assigns the current_value as both the initial EWMA value and previous_ewma_value, then returns it. For subsequent calculations, the algorithm calculates the new EWMA value, assigns the result to previous_ewma_value for future reference, and returns the new EWMA value.

Threshold Calculation

The threshold calculation component determines the Upper Control Limit (UCL) and Lower Control Limit (LCL) using Equation (6).

$$LCL = CL - f * \sigma_z$$
$$UCL = CL + f * \sigma_z \quad (6)$$

Where the factor f is set to be equal to 3-sigma control limits and σ_z is the standard deviation of EWMA values. The threshold calculation algorithm, outlined in Algorithm 4.2, starts by determining the CL (Center Line), which is the mean of the EWMA values of each feature. This mean value signifies the expected network behavior under standard conditions. Following this, the algorithm calculates the standard deviation of these EWMA values. The standard deviation is essential as it measures the amount of variation or spread within the dataset, ensuring that the control limits are appropriately scaled according to the data's inherent variability. Subsequently, the UCL and LCL are computed. The LCL and UCL of each feature serve as threshold values in the EWMA-Based jamming detection component. The component continuously monitors the EWMA values of incoming data. If these values surpass the UCL or dip below the LCL, it indicates potentially a jamming attack.

Algorithm 4.2: Threshold Calculation

```
Input: EWMA_NORMAL_DF //Dataset obtained after EWMA calculation
Output: Thresholds
Begin
    Features = ['RSSI','PER']
    Thresholds = {}
    For feature in Features
        Mean = EWMA_NORMAL_DF[feature].mean()
        STD = EWMA_NORMALA_DF[feature].std()
        UCL = Mean + 1.5 * STD // upper control limit
        LCL = Mean - 1.5 * STD // lower control limit
        Thresholds[feature] = (UCL, LCL)
        Return Thresholds
    End For
End
```

Model Training

For jamming attack type classification, we have chosen Random Forest classifier. In the literature Random Forest is being the most accurate machine learning for jamming attack detection [10, 13, 15]. The model training component of the proposed solution, outlined in Algorithm 4.3, is responsible for training the Random Forest classifier using the EWMA-smoothed training dataset. This component utilizes the features extracted using EWMA and their corresponding jamming attack types.

Algorithm 4.3: Model Training

```
Input:
    X_train, y_train // Training Dataset
    N_estimators
    Max_depth
    Min_samples_split
Output:
    RF_classifier // Trained Random Forest Classifier Model
Begin
    RF_classifier = RandomForestClassifier(n_estimators,
                                           max_depth,
                                           min_samples_split)
    RF_classifier.fit(X_train, y_train)
    Return RF_classifier
End
```

The model training component employs the RandomForestClassifier from the scikit-learn library to train the classifier on the prepared dataset. This is achieved through the application of the *fit()* method, which serves as a training mechanism inherent to the instances of the RandomForestClassifier class. During this process, each decision tree in the forest is trained on a random subset of the training data and a random selection of features. This randomness helps to ensure that the individual trees are decorrelated, reducing the risk of overfitting and improving the generalization performance of the classifier.

The performance of the Random Forest classifier can be optimized by adjusting its hyperparameters, including the number of trees in the forest (n_estimators), the maximum depth of the trees (max_depth), and the minimum number of samples needed to split an internal node (min_samples_split). This parameter tuning helps improve the accuracy of the trained model. The parameters are chosen in a way that minimizes computational overhead while ensuring they are suitable for implementation within the TinyML framework. By training the Random Forest classifier on the EWMA-smoothed training dataset, the proposed solution aims to achieve accurate classification of jamming attack types.

4.3.2 Testing Phase

EWMA-Based Jamming Detection

The EWMA-Based jamming detection components determine whether jamming attack is occurring or not using the EWMA-smoothed dataset from the output of the EWMA computation component. The EWMA Computation algorithm has already been discussed in the training phase section. The EWMA-Based Jamming Detection component determines whether a jamming attack is occurring or not. It does this by comparing the EWMA value of the input features against the UCL and LCL for each feature value. The detailed algorithm for detecting jamming attacks using EWMA is outlined in Algorithm 4.4.

Algorithm 4.4: EWMA-Based Jamming Detection

```
Input: EWMA_RSSI, EWMA_PER // Input feature value after EWMA calculation
Output: Jamming_Detected // True if detected or False if not

Begin
    Jamming_Detected = False // Initialize the variable
    If ( LCL_RSSI > EWMA_RSSI OR EWMA_RSSI > UCL_RSSI OR
         LCL_PER > EWMA_PER OR EWMA_PER > UCL_PER )
       Jamming_Detected = True
    End If
    Return Jamming_Detected
End
```

Jamming Classification

Once the EWMA-Based jamming detection component determines that a jamming attack is occurring, the system transitions to the jamming classification phase. This phase employs a trained Random Forest Classifier model to identify the specific type of jamming attack, which is crucial for understanding the nature of the attack and implementing appropriate countermeasures. The algorithm for the jamming classification component is outlined in Algorithm 4.5. The types of jamming attacks to be classified include constant, reactive, and periodic jamming. Constant jamming involves the attacker continuously transmitting radio signals to congest the communication channel, preventing legitimate sensor nodes from transmitting or receiving data. Reactive jamming occurs when the attacker transmits jamming signals only when it detects activity on the communication channel, making it more selective and harder to detect. Periodic jamming involves intermittent transmission of jamming signals at regular intervals, causing occasional disruptions in communication. The importance of jamming classification lies in its ability to provide detailed insights into the nature of the jamming attack, which is essential for implementing appropriate countermeasures [8].

Algorithm 4.5: Jamming Classification

```
Input:
    X_test // Features of the test dataset
    RF_classifier // Trained Random Forest Classifier Model
Output:
    y_pred // Predicted labels for the test dataset
Begin
    y_pred = RF_classifier.predict(X_test)
    Return y_pred
End
```

The proposed solution for detecting and classifying jamming attacks is clearly outlined in Algorithm 4.6. The algorithm follows a structured process that begins with the computation of EWMA for two key input features: RSSI and PER. These smoothed values are then used to identify potential jamming attacks, and, if an attack is detected, the algorithm proceeds to classify the type of jamming attack using a Random Forest classifier. In the first step, the EWMA method is applied to RSSI and PER values. This technique is essential for smoothing out random variations and highlighting patterns in the data, making it easier to detect anomalies caused by jamming attacks. The EWMA approach assigns more weight to recent data, which helps the system quickly identify sudden changes in network features. After the EWMA values are computed, the algorithm uses these smoothed metrics to detect jamming attacks. By comparing the EWMA values to predefined thresholds, it determines whether the network behavior deviates significantly from normal conditions. This step ensures that the system is sensitive enough to detect real attacks while minimizing false alarms caused by natural network fluctuations. If a jamming attack is detected, the next step is to classify the type of attack. The Random Forest classifier is used for this task, as it is well-suited for handling complex data relationships and is robust against noise. Using the EWMA-smoothened RSSI and PER values, the classifier identifies the jamming attack type, such as constant, reactive, or deceptive jamming. This classification provides useful information for understanding the nature of the attack and taking appropriate countermeasures. The combination of EWMA for feature smoothing and Random Forest for classification makes this solution both efficient and accurate. EWMA reduces the effect of random changes in the network, enabling the system to focus on meaningful deviations. Meanwhile, the Random Forest classifier ensures reliable and accurate classification of jamming attacks. Together, these methods create an accurate and lightweight approach for detecting and classifying jamming attacks in WSN.

Algorithm 4.6: Jamming Attack Detection and Classification Algorithm

```
Input: RSSI, PER
Output: Jamming_Detected,Jamming_Type
Begin
    EWMA_RSSI = EWMA_Computation(RSSI)
    EWMA_PER = EWMA_Computation(PER)
    Jamming_Detected = EWMA-Based_Jamming_Detection (EWMA_RSSI, EWMA_PER)
    If Jamming_Detected
        Jamming_Type = Jamming_Classification (EWMA_RSSI, EWMA_PER)
        Return Jamming_Type
    Else
        Continue
    End If
End
```

4.4 Summary

In this chapter, we have outlined the proposed solution for detecting and classifying jamming attacks in WSNs. The solution is structured around several key components designed to work together to detect and classify jamming attacks effectively.

We began by discussing the design considerations, which highlighted the importance of various factors such as jamming attack scenarios, feature selection, the use of the EWMA for jamming detection, and the selection of the Random Forest classifier for attack classification. Each of these elements was chosen to improve the proposed method accuracy while minimizing false alarm and overhead. The EWMA computation component was described, emphasizing its role in smoothing time-series data and removing noise, thus enhancing the accuracy of jamming detection. The calculation and combination of EWMA values for RSSI and PER were outlined, demonstrating how these metrics contribute to a comprehensive detection mechanism.

Finally, the two-level jamming detection and classification design was discussed, justifying the use of a Random Forest classifier due to its proven accuracy in previous studies. This approach is advantageous in reducing false alarms and improving overall detection performance. The proposed solution combines EWMA and Random Forest, utilizing the strengths of each to overcome their individual limitations for effective jamming attack detection and classification in WSN. This two-level method utilizes EWMA to reduce the computational overhead associated with Random Forest classifier, while the Random Forest enhances the accuracy of detection and classification.

Chapter Five: Experimentations and Evaluation

5.1 Introduction

In this Chapter, we present the experimentation and evaluation of the proposed solution for jamming attack detection and classification in WSNs. We describe the experimental setup used for evaluation, the evaluation metrics employed, and the results obtained from the experiments conducted. The primary objective of this chapter is to assess the effectiveness and robustness of the proposed solution in accurately detecting and classifying jamming attacks.

By conducting comprehensive experiments and evaluations, we aim to validate the performance of the solution and provide insights into its strengths and limitations. We present the experimental results obtained from our evaluation. This involves assessing the performance of the Random Forest classifier using evaluation metrics such as accuracy, precision, recall, and F1-score. Additionally, we visualize the results using techniques such as confusion matrices. In the result and discussion section, we interpret and analyze the experimental results, highlighting the strengths and weaknesses of the proposed solution. We compare our results with existing methods and provide insights gained from the experiments.

5.2 Dataset

For evaluation purposes, mitigating RF jamming attacks at the physical layer with machine learning dataset from experimental evaluations conducted by Jacovic *et al.* [13] is used in this study. This dataset comprises detailed logs of features recorded during various jamming attack scenarios, as well as scenarios representing normal network operation, network congested links, and weak link conditions. The dataset provides a diverse range of scenarios, allowing for comprehensive evaluation of the proposed solution's effectiveness in detecting and classifying jamming attacks in WSNs. The data preparation process involved organizing and structuring the raw log files to create training and testing datasets suitable for the implementation of the proposed method. The raw data consisted of four log files, each representing a distinct scenario: normal traffic, constant jamming, periodic jamming, and reactive jamming. Each log file is added into the dataset and labeled according to its respective scenario type to facilitate classification. The labeled data from the four scenarios was then combined into a single dataset for uniform processing and analysis.

Feature Selection

The feature selection process involved the extraction of relevant features from the dataset for jamming attack detection and classification. The Dataset contains three features: RSSI, EVM, and PER. The features selected from the dataset include RSSI and PER. These features were selected based on their significance in jamming attack detection and their availability within the dataset. We haven't used EVM as a feature due to the computational complexity, hence it is not applicable to resource constrained WSNs. Whereas RSSI is directly obtained from the transceiver and PER is calculated with limited overhead.

Training/Testing Data Split

To ensure balanced training and testing datasets, the dataset was divided into two distinct sets: a training set and a testing set, utilizing a 70% to 30% ratio, respectively. Table 5.1 shows the number of records used in training set and testing set.

Table 5.1: Number of records used in dataset

Class	Number of records used in dataset	
	Training set (70%)	**Testing set (30%)**
Normal	3,294	1,412
Constant Jamming	927	398
Periodic Jamming	921	395
Reactive Jamming	821	352
Sum	**5,963**	**2,557**

The training dataset undergoes processing through the EWMA computation to generate a new dataset with EWMA values. This processed dataset, containing EWMA values, is utilized for training the Random Forest classifier. On the other hand, the testing set remains unchanged, serving as a real-world input for the testing phase of the proposed solution.

The EWMA computation component is a crucial aspect of our proposed method for noise removal in the dataset. Wireless networks are inherently susceptible to unintentional interference, commonly referred to as noise. This noise can stem from various sources, including environmental factors such as lightning and crosstalk due to nearby electromagnetic devices. Given the inevitable presence of unintentional

interference in wireless networks, it is vital for a jamming attack detection system to effectively differentiate between these types of noise and actual jamming attacks.

Data smoothing techniques like EWMA are beneficial in filtering out the impact of unintentional interference, thereby allowing the system to focus on identifying jamming attacks. EWMA helps by smoothing the values affected by unintentional interference and enhances the clarity of the data. This computation step plays a significant role in improving the detection and classification of jamming attacks by providing a clearer distinction between different jamming classes.

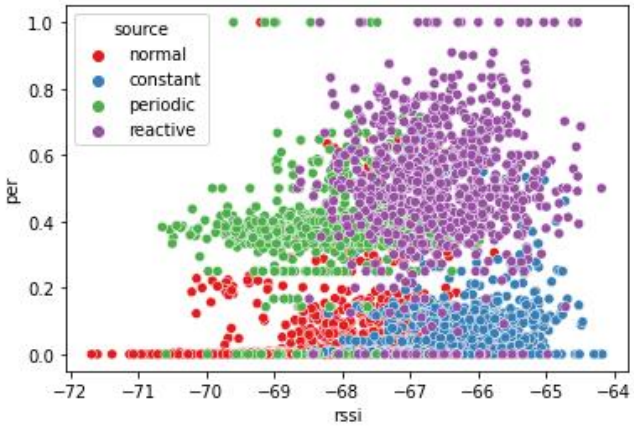


Figure 5.1: Original dataset visualization

In Figure 5.1, we present the visualization of the original dataset without EWMA computation being applied. The plot demonstrates considerable overlap between different jamming classes, which complicates the detection and classification of jamming attacks. Specifically, the classes normal and constant, and periodic and reactive are not distinctly separable, resulting in potential challenges for any machine learning algorithm attempting to classify these instances accurately.

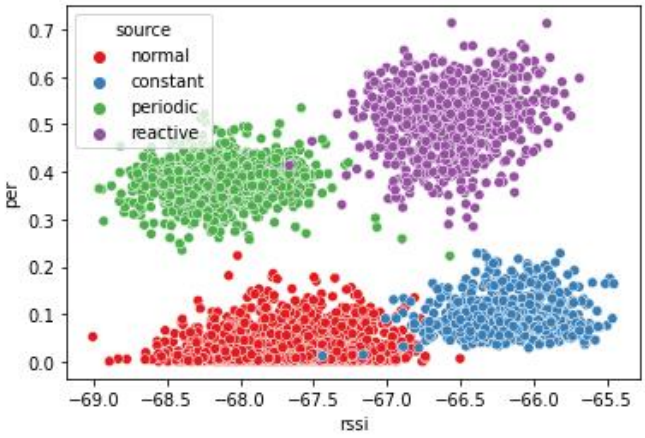


Figure 5.2: Dataset visualization after EWMA computation

However, after applying the EWMA computation, as illustrated in Figure 5.2, the separation between different jamming classes becomes significantly more pronounced. The clusters for each class normal, constant, periodic, and reactive are now clearly distinguishable. This enhanced separation is critical as it directly contributes to the improved performance of jamming attack detection and classification algorithms. By reducing the noise and highlighting the inherent patterns within the data, EWMA-smoothed data facilitates more accurate and reliable detection of jamming attacks.

Using the output of the EWMA computation on the training dataset, we have calculated the threshold for each feature value and stored it in the threshold array. The thresholds are determined by analyzing the statistical properties of the EWMA-smoothed data, including the upper and lower control limits, which are derived based on a 3-sigma approach. These control limits ensure that normal variations in the data do not trigger false positives, while also providing a robust baseline for anomaly detection. During the experimentation phase, we observed that carefully tuned thresholds significantly reduced the false positive rate without compromising detection accuracy.

5.3 Experimental Setup

In this study, all experiments and evaluations were conducted on an HP ENVY Laptop 13-ba1093cl with Windows 11 Home (23H2), an 11th Gen Intel Core i5-1135G7 processor (2.40 GHz), 16 GB of RAM, and a 500 GB SSD

We utilized Python version 3.1.1 from Anaconda alongside a variety of libraries, as detailed in Table 5.2. Anaconda not only provides the Python interpreter but also incorporates numerous beneficial libraries and the Spyder IDE.

Table 5.2: Software Libraries Used

Library Name	Usage	Version
Pandas	Reading and Writing CSV files	V2.1.4
SciKit-Learn	Machine Learning Classifier and Metrics	V1.4.2
Matplotlib	For Visualizing Data	V3.8.1
Seaborn	For making statistical graphics	V0.13.1

5.4 Experiments and Results

In this section, we present experiments conducted and analyze the results of our proposed solution for detecting and classifying jamming attacks in WSNs. The evaluation metrics used to assess the performance include detection rate, accuracy, precision, recall, and F1-score.

5.4.1 Evaluation Metrics

In this section, we discuss the evaluation metrics used to assess the performance of the proposed solution for jamming attack detection and classification in WSNs.

Confusion Matrix

A confusion matrix is a crucial evaluation tool in machine learning, particularly for classification tasks, as it provides a detailed breakdown of model performance. It is a table that contrasts the actual target values with the predictions made by the model, categorizing them into four types: True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN). This matrix enables the calculation of various performance metrics such as accuracy, precision, recall, and F1 score.

Accuracy

Accuracy is a key metric for assessing the overall performance of a classification model. It indicates the ratio of correctly predicted instances to the total number of instances in the dataset. Mathematically, accuracy is calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (7)$$

Where:

- TP (True Positive) refers to the count of instances accurately identified as positive.
- TN (True Negative) refers to the count of instances accurately identified as negative.
- FP (False Positive) refers to the count of instances wrongly identified as positive.
- FN (False Negative) refers to the count of instances wrongly identified as negative.

A higher accuracy value shows the effectiveness of the classifier in correctly identifying both positive and negative instances.

Precision

Precision evaluates the correctness of the positive predictions made by the classifier. It is defined by the ratio of true positive instances to the total number of instances that were predicted to be positive. Mathematically, precision is calculated as:

$$Precision = \frac{TP}{TP + FP} \quad (8)$$

Precision offers insights into how well the classifier avoids false positives. A higher precision score signifies a reduced occurrence of false positive predictions.

Recall

Recall, also known as the detection rate, evaluates the classifier's ability to accurately identify positive instances. It is determined by the ratio of true positive instances to the total number of actual positive instances. Mathematically, recall is calculated as:

$$Recall = \frac{TP}{TP + FN} \quad (9)$$

Recall offers insights into how well the classifier avoids false negatives. A higher recall value signifies a reduced occurrence of false negative predictions.

F1-score

The F1-score, which is the harmonic mean of precision and recall, offers a balanced evaluation of how well the classifier performs. Mathematically, it is calculated as:

$$F1 - score = 2\,x\,\frac{Precission\ X\ Recall}{Precission +\ Recall} \quad (10)$$

The F1-score ranges from 0 to 1, with higher values indicating better performance. It's especially valuable for imbalanced datasets since it considers both false positives and false negatives.

By evaluating the proposed solution using these metrics, we can gain insights into its effectiveness and robustness in detecting and classifying jamming attacks in WSNs. Additionally, these metrics help in identifying areas for improvement and fine-tuning the solution for optimal performance.

5.4.2 EWMA-Based Jamming Detection

The jamming detection component, discussed in detail in Chapter Four, serves as the first level of the proposed jamming attack detection and classification method. Utilizing the EWMA control charts, this component is designed to identify deviations from normal network behavior, which may indicate the presence of a jamming attack. For evaluation, we applied the jamming detection component to the test dataset and measured its detection rate. The detailed results are presented in Table 5.3.

Table 5.3: EWMA-Based Jamming Detection Result

Class	Correctly detected	Miss detected	Detection rate
Constant	397	1	99.75%
Periodic	395	0	100%
Reactive	352	0	100%
Normal	1137	275	80.52%

The results indicate that the EWMA-Based method effectively identifies jamming attack. We achieved a high detection rate of 99.91%. Which improves the jamming detection result obtained without utilizing EWMA for smoothing, detection rate of 84.72, shown in Table 5.4.

Table 5.4: Results of Jamming Detection without EWMA

Class	Correctly detected	Miss detected	Detection rate
Constant	246	152	61.81%
Periodic	386	9	97.72%
Reactive	338	14	96.02%
Normal	1162	250	82.29%

However, the results also revealed a high false positive rate of 19.48% compared to the result without EWMA 17.71%. Since the proposed solution includes an attack classification stage following jamming detection, these false positives will be filtered out by the classification model. This additional step enhances the overall detection accuracy of the system.

We evaluated the jamming detection component in terms of detection rate and false positive rate across various alpha values. The results of our evaluation are shown in Figure 5.3.

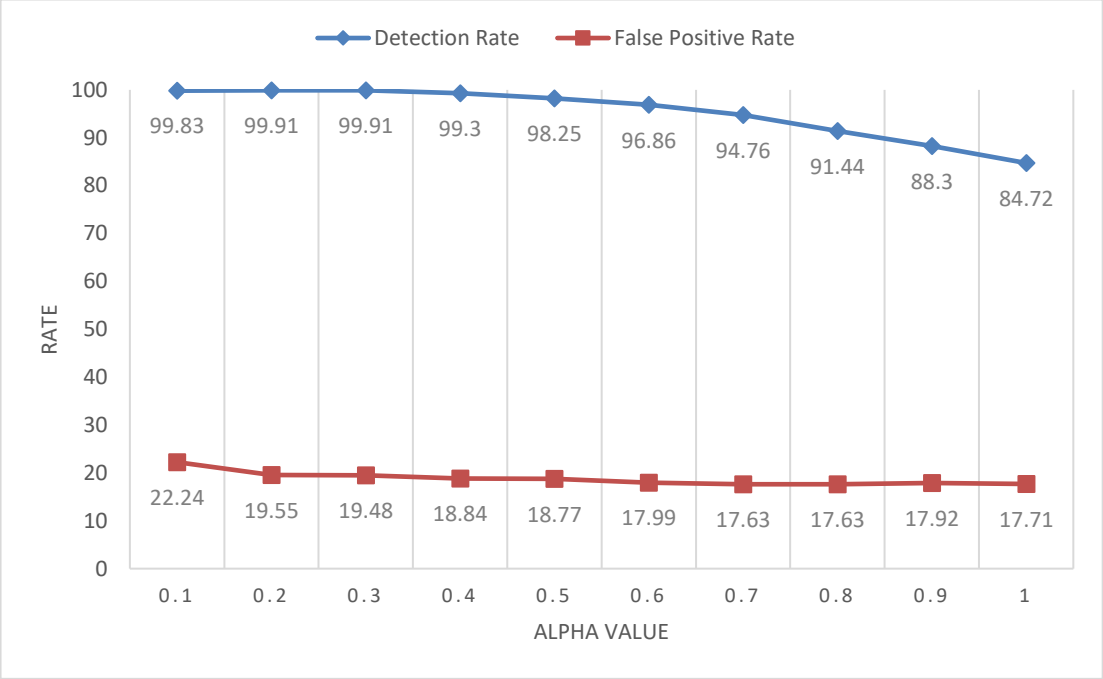


Figure 5.3: Comparison of results for different alpha values of EWMA

The graph in Figure 5.3 illustrates the effect of different alpha values on the detection rate and false positive rate. When using lower alpha values, the system gives more weight to historical data, resulting in a consistently high detection rate and a relatively stable false positive rate. This indicates that using lower alpha value is highly effective at detecting both subtle and persistent jamming attacks, as it considers long-term trends. However, as the alpha value increases the detection rate starts to decline significantly. This is because higher alpha values prioritize recent data, making it more responsive to short-term fluctuations but less sensitive to gradual changes, which may lead to undetected attacks. Although the false positive rate decreases slightly with higher alpha values, the trade-off in detection performance suggests that lower alpha value of 0.3 provide the optimal balance for accurate jamming attack detection.

We also evaluated the jamming attack detection rate and false positive rate using each feature individually and in combination. The results, shown in Figure 5.4, indicate that relying on a single detection metrics is insufficient for accurately detecting jamming attacks. When detection metrics were used separately, the detection rates were lower, suggesting that single detection metrics do not provide a comprehensive view. However, when PER used together with RSSI, the detection accuracy significantly improved. This demonstrates the importance of a multi-feature approach in effectively detecting jamming attacks [6, 10].

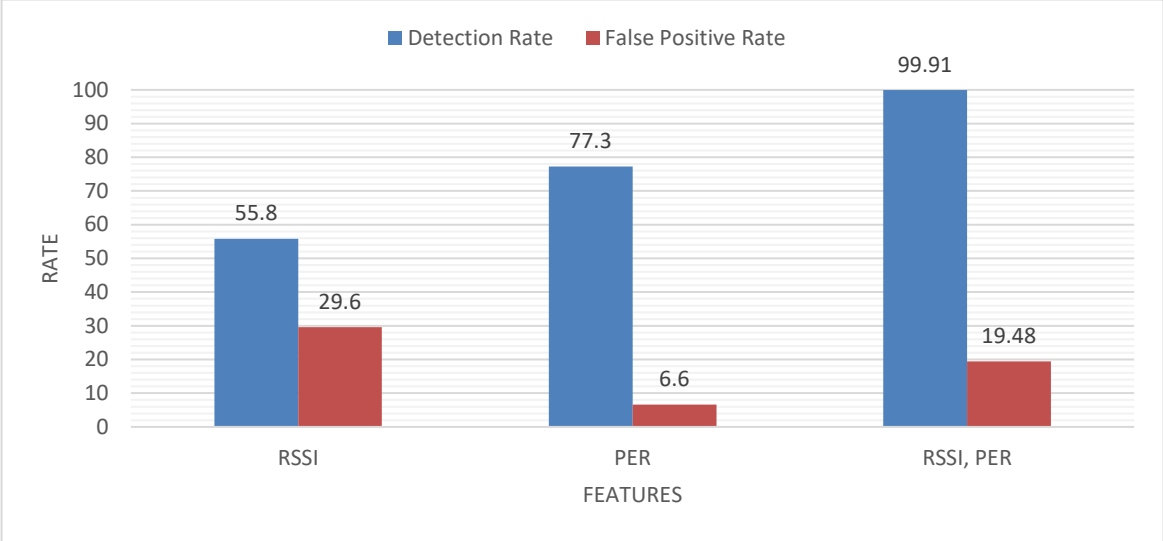
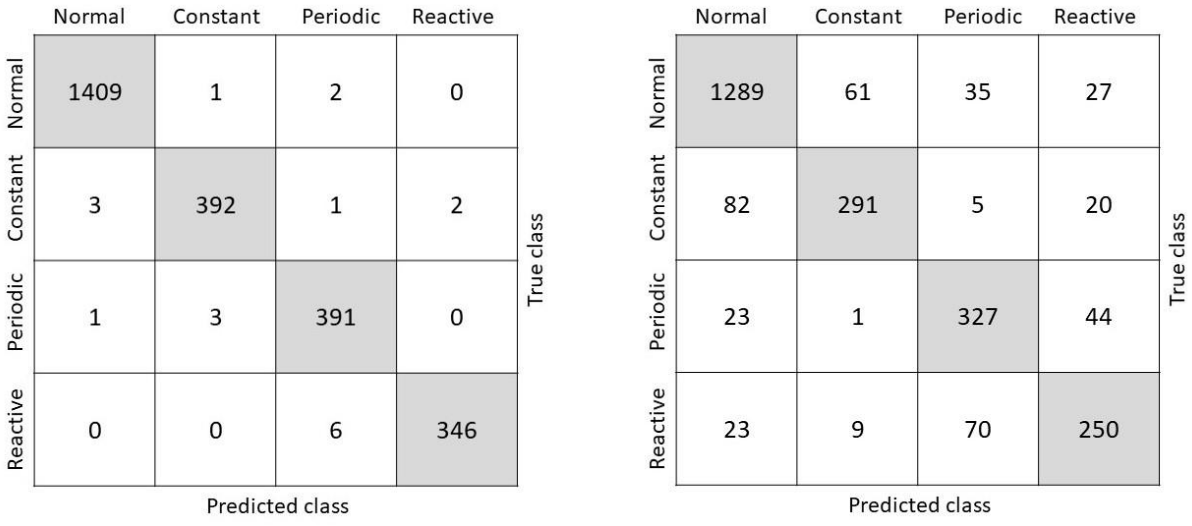


Figure 5.4: Comparison of results by using different detection metrics

5.4.3 Jamming Classification: Random Forest Classifier

In the jamming classification stage, we use a Random Forest classifier to identify the specific type of jamming attack once a jamming attack has been detected. This research tries to classify the common jamming attack types these are constant, periodic and reactive. To assess the effectiveness of this classifier, we measure key performance metrics, including precision, recall, and F1-score from the confusion matrix shown in Figure 5.5.



(a) With EWMA processing (b) Without EWMA processing

Figure 5.5: Confusion matrices of the Random Forest classifier

The results of our proposed approach, which incorporates EWMA computation, are presented in Table 5.5. This approach achieved a detection accuracy of 99.26%.

Table 5.5: Results of Random Forest Classifier with EWMA

Class	Precision	Recall	F1-Score	Support
Constant	0.99	0.98	0.99	398
Periodic	0.98	0.99	0.98	395
Reactive	0.99	0.98	0.99	352
Normal	1.0	1.0	1.0	1412
Weighted Avg	**0.99**	**0.99**	**0.99**	**2557**

We compare the classifier's performance on datasets with and without EWMA. The results without EWMA computation, using the original dataset, are presented in Table 5.6.

Table 5.6: Results of Random Forest Classifier without EWMA

Class	Precision	Recall	F1-Score	Support
Constant	0.80	0.73	0.76	398
Periodic	0.74	0.82	0.78	395
Reactive	0.73	0.71	0.72	352
Normal	0.91	0.91	0.91	1412
Weighted Avg	**0.84**	**0.84**	**0.84**	**2557**

The results demonstrate a significant improvement in the classifier's ability to accurately identify different jamming attack types when EWMA computation is applied. The EWMA computation step smooths the data, reduces noise, and enhances the quality of the features used for classification.

We further evaluated our results by comparing them with other data smoothing techniques, which are commonly used in the literature to reduce noise in data while preserving underlying patterns. These techniques include simple moving average (SMA), low pass filter (LPF), and median filter (MF). Figure 5.6 illustrates the comparison of results, highlighting the performance of the EWMA method. Among the evaluated techniques, EWMA achieved superior accuracy. This suggests that EWMA is

more effective in capturing the underlying patterns in the data, especially when dealing with noisy environments, making it a preferable choice for jamming attack detection in WSNs.

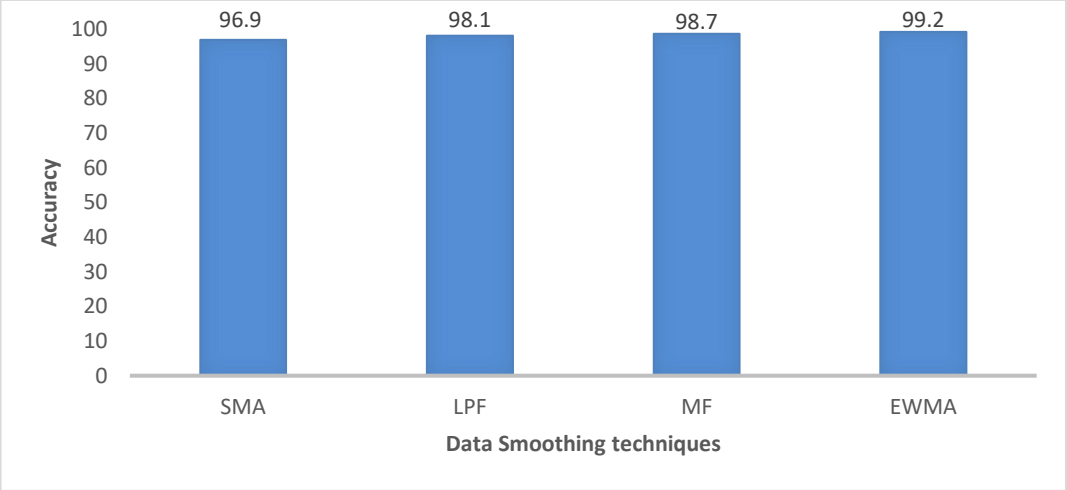


Figure 5.6: Comparison of different data smoothing techniques

The computational overhead of the proposed method is evaluated in terms of testing time and memory allocation, comparing it with approaches that utilize only EWMA or Random Forest classifiers. As shown in Table 5.7, the EWMA-based method achieves an average testing time per instance of approximately 0.0068 milliseconds with an average memory allocation of 0.4195 KB, indicating low computational overhead. In contrast, the Random Forest method incurs significantly higher computational costs, with an average testing time per instance of 0.9530 milliseconds and an average memory allocation of 11.43 KB. In this experiment, we demonstrate that statistical-based methods, such as EWMA, inherently have lower computational overhead compared to machine learning-based methods, such as Random Forest, as stated in the literature.

Our proposed approach effectively balances the tradeoff between jamming detection accuracy and computational overhead by combining these two methods in a hierarchical two-level architecture. In this setup, the Random Forest classifier is triggered only when the EWMA-based jamming detection mechanism flags a potential jamming attack. Since jamming attacks occur sporadically, this approach ensures that the Random Forest classifier run only when potential jamming attack is detected, which minimizes computational overhead while maintaining high accuracy. The computational efficiency of the system largely depends on whether a jamming attack is occurring. In the absence of a jamming attack, the system remains at the EWMA-based jamming detection level, resulting in a computational load comparable to that of the standalone EWMA method. This addresses the limitations of machine learning-

based approaches by avoiding unnecessary computational costs when no attacks are detected. Conversely, when a potential jamming attack is flagged by the EWMA-based jamming detection, the system transitions to the Random Forest classifier. This step not only classifies the type of attack but also significantly reduces false positives, addressing the limitation of EWMA in terms of accuracy.

Table 5.7: Comparison of Computational Overhead Metrics

Metrics	**EWMA**	**RF**	**EWMA-RF**
Total Testing Time	≈ 19.57 ms	≈ 2.4367 s	≈ 1.634 s
Average Testing Time/ Instance	≈ 0.0068 ms	≈ 0.9530 ms	≈ 0.6391 ms
Total Memory Usage	≈ 1.07 MB	≈ 29.23 MB	≈ 20.08 MB
Average Memory Allocation/ Instance	≈ 0.4195 KB	≈ 11.43 KB	≈ 7.85 KB

5.5 Discussion

Our experiments demonstrate the effectiveness of combining EWMA-based detection with Random Forest classification in detecting and classifying jamming attacks in WSNs. The application of EWMA significantly improved detection accuracy, reducing noise and enhancing the performance of the classification model. Specifically, the EWMA-based detection method achieved a 99.91% detection rate across constant, periodic, and reactive jamming attacks, compared to 84.72% without EWMA, on the original dataset. This improvement in detection rate was further reflected in the higher accuracy, precision and recall values achieved in the classification stage, where the Random Forest model obtained an accuracy of 99.26% when EWMA computation was applied. By integrating EWMA for jamming detection and Random Forest for attack classification, our method provides a balanced solution that is both highly accurate and computationally feasible, making it well-suited for real-time applications in resource-constrained WSN environments.

We compared the performance of our method with existing approaches in the literature to assess its effectiveness. Our proposed solution for jamming attack detection demonstrates a lower jamming detection rate compared to the method presented by Osaniye *et al.* [5], where the false positive rate, a critical factor in determining overall accuracy, is not provided. However, in terms of jamming attack classification, our approach shows significant improvements in accuracy when compared to other

closely related methods. The results, summarized in Table 5.8, indicate that our method achieves high classification accuracy, outperforming existing approaches. The integration of the EWMA computation step has notably enhanced detection and classification performance by reducing noise and improving feature quality. This comparison highlights the effectiveness of our proposed method in addressing the challenges posed by jamming attacks in WSNs.

Table 5.8: Summary of Comparison with Existing Methods

Year	Paper	Method Used	Detection Metrics	Result
2018	Osaniye *et al.* [5]	EWMA	Packet Inter-arrival Time (IAT)	For ≥ 20 jammed packets: 100%
2019	Upadhyaya *et al.* [11]	Random Forest with AdaBoost	RSSI with multi-path profile data	99.01% accuracy
2020	Arjoune *et al.* [15]	Random Forest	Bad Packet Ratio (BPR), PDR, RSSI, Clear Chanel Assessment (CCA)	96.6% accuracy
2022	Jacovic *et al.* [13]	Random Forest	RSSI, PER, EVM	95% detection rate 93% accuracy
2023	Testi *et al.* [12]	Neural Network	PDR, Signal to Noise ratio (SNR)	99% accuracy
2024	This Paper	EWMA and Random Forest	RSSI, PER	99.91% detection rate and 99.26% accuracy

Chapter Six: Conclusion and Future Work

This Chapter summarizes the key findings of this research, highlighting the effectiveness and accuracy of the proposed jamming attack detection and classification method. Additionally, it discusses the implications of these findings and their potential applications in various anomaly detection scenarios. Finally, the chapter outlines future research directions to further expand upon the current work.

6.1 Conclusion

In this thesis, we have presented a novel approach to jamming attack detection and classification in WSNs, combining EWMA for jamming attack detection with a Random Forest classifier for jamming attack classification. The research was motivated by the critical need to safeguard WSNs, which are increasingly deployed in sensitive and mission-critical applications such as healthcare, environmental monitoring, industrial automation, smart agriculture, smart cities, and military defense. The proposed solution aims to address the challenges of accurately detecting and classifying jamming attacks to ensure the security of WSNs.

We began by providing a comprehensive overview of WSNs, highlighting their architecture, communication infrastructure, and various network topologies. The literature review covered existing detection metrics used for detection of jamming, laying the groundwork for the proposed solution. We then detailed the design of the proposed jamming attack detection and classification method, emphasizing the selection of key features such as RSSI and PER, and the implementation of the EWMA for jamming detection and Random Forest classifier model for classification.

We evaluated our approach and demonstrated that it achieves a high accuracy of 99.26% in jamming attack classification and a 99.91% detection rate for jamming attack detection. The experimental results indicate that incorporating EWMA significantly enhances classification accuracy in the context of jamming attacks. This improvement suggests potential applicability to other anomaly detection and classification tasks, especially in environments where noise is prevalent. Notably, our proposed method incurs no additional overhead on the sensor nodes.

6.2 Contribution

The main contributions of this work are the following:

- We propose a novel two-level detection system that integrates EWMA for jamming detection and a Random Forest classifier for classifying the types of jamming attacks.
- We leverage EWMA to process raw input data, effectively smoothing out noise while retaining essential features necessary for attack detection and classification.
- We have chosen two essential features, RSSI and PER, that are readily available on existing platforms through existing hardware and cannot affect the original communication.
- We address a key challenge in WSNs, optimizing the balance between detection accuracy and computational overhead.

6.3 Future Work

While this research attempts to address the gap in the detection and classification of jamming attacks in WSNs, it leaves room for further exploration in subsequent stages such as localization and mitigation of these attacks. Future work could also focus on implementing TinyML techniques and evaluating its performance in terms of detection time, memory usage, and energy consumption.

References

[1] S. Jaitly, H. Malhotra and B. Bhushan, "Security Vulnerabilities and Countermeasures against Jamming Attacks in Wireless Sensor Networks: A Survey," 2017 International Conference on Computer, Communications and Electronics (Comptelix), 2017.

[2] A. Muhammed, M. Almaiah, A. Alsayed, and O. Almomani, "An Anonymous Channel Categorization Scheme of Edge Nodes to Detect Jamming Attacks in Wireless Sensor Networks," Sesnors 20, No. 8, 2020.

[3] J. Zhang, Y. Hu, and H. Li, "Research on Wireless Sensor Network Positioning Based on Genetic Algorithm," Wireless Communications and Mobile Computing, 2021.

[4] M. Jeyaselvi, S. Suchitra, M. Sathya, and R. Mekala, "Energy Efficient Witness Based Clone and Jamming Attack Detection in WSN," in Journal of Green Engineering (JGE), Vol. 11, No. 2, 2021.

[5] O. Osanaiye, A. S. Alfa, and G. P. Hancke, "A Statistical Approach to Detect Jamming Attacks in Wireless Sensor Networks," Sensors 18, No. 6, 2018.

[6] S. Godala and R. P. V. Vaddella, "A Study on Intrusion Detection System in Wireless Sensor Networks," International Journal of Communication Networks and Information Security, Vol. 12, No. 1, pp. 127–141, 2020.

[7] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing, pp. 46–57, May 2005.

[8] H. Pirayesh and H. Zeng, "Jamming Attacks and Anti-Jamming Strategies in Wireless Networks: A Comprehensive Survey," IEEE Communications Surveys & Tutorials, Vol. 24, No. 2, pp. 767-809, 2022.

[9] M. Cheng, Y. Ling, and W. B. Wu, "Time Series Analysis for Jamming Attack Detection in Wireless Networks," IEEE Global Communications Conference, pp. 1-7, 2017.

[10] Z. Feng and C. Hua, "Machine Learning-based RF Jamming Detection in Wireless Networks," 2018 Third International Conference on Security of Smart Cities, Industrial Control System and Communications (SSIC), pp. 1-6, 2018.

[11] B. Upadhyaya, S. Sun and B. Sikdar, "Machine Learning-based Jamming Detection in Wireless IoT Networks," Asia Pacific Wireless Communications Symposium (APWCS), 2019.

[12] E. Testi, L. Arcangeloni, and A. Giorgetti, "Machine Learning-Based Jamming Detection and Classification in Wireless Networks," ACM Workshop on Wireless Security and Machine Learning, pp. 39-44, 2023.

[13] M. Jacovic, R. Rey, G. Mainland, and R. Dandekar, "Mitigating RF Jamming Attacks at the Physical Layer with Machine Learning," IET Communications 17, No. 1, pp. 12-28, 2022.

[14] G.S Kasturi, A. Jain, and J. Singh, "Detection and Classification of Radio Frequency Jamming," In Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, Vol. 11, 2020.

[15] Y. Arjoune, F. Salahdine, M. S. Islam, E. Ghribi, and N. Kaabouch, "A Novel Jamming Attacks Detection Approach Based on Machine Learning for Wireless Communication," 2020 International Conference on Information Networking (ICOIN), pp. 459-464, 2020.

[16] C. Greco, P. Pace, S. Basagni, and G. Fortino, "Jamming detection at the edge of drone networks using Multi-layer Perceptrons and Decision Trees," Appied Soft Computing, Vol. 111, 2021.

[17] A. Christian, A. Adamou, M. Gueroui, and Z. Aliouat, "Big Data Collection in Large-Scale Wireless Sensor Networks," Sesnors 18, No. 18, 2018.

[18] S. R. Jino Ramson and D. J. Moni, "Applications of Wireless Sensor Networks — A survey," 2017 International Conference on Innovations in Electrical, Electronics, Instrumentation and Media Technology (ICEEIMT), pp. 325-329, 2017.

[19] C. Mallick and S. Satpathy, "Challenges and Design Goals of Wireless Sensor Networks," in International Journal of Computer Applications, Vol. 179, No. 28, 2018.

[20] S. Diaz, D. Mendez, and R. Kraemer, "A Review on Self-Healing and Self-Organizing Techniques for Wireless Sensor Networks, " in Journal of Circuits, Systems and Computers, Vol. 28, No. 5, 2019.

[21] W. B. Heinzelman, A. P. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," in IEEE Transactions on Wireless Communications, Vol. 1, No. 4, pp. 660-670, 2002.

[22] L. Karim, N. Nasser, H. Abdulsalam and I. Moukadem, "An Efficient Data Aggregation Approach for Large Scale Wireless Sensor Networks," 2010 IEEE Global Telecommunications Conference GLOBECOM, pp. 1-6, 2010.

[23] J. Li, P. Zhao, Y. Sun, H. Yuan, and H. Xu, "Research and Application of Energy-Efficient Management Approach for Wireless Sensor Networks," in Sensors 23, Vol. 23, No. 3, 2023.

[24] S. Shukry, "Stable Routing and Energy-Conserved Data Transmission over Wireless Sensor Networks," in EURASIP Journal on Wireless Communications and Networking, Vol. 36, No. 1, 2021.

[25] S. Anand and T. N. Gowthami, "Cluster-Based Energy Efficient Protocol for Wireless Sensor Network," 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), 2020.

[26] F. Afroz and B. Robin, "Energy-efficient MAC Protocols for Wireless Sensor Networks: A Survey," in International Journal of Sensor Networks, Vol. 32, No. 3, 2020.

[27] P. Truong, T. Hai, and N. Tram. "A Reconfigurable Hardware Platform for Low-Power Wide-Area Wireless Sensor Networks," in Journal of Physics: Conference Series, Vol. 1432, No. 1, 2020.

[28] A. J. Williams, M. F. Torquato, I. M. Cameron, A. A. Fahmy, and J. Sienz, "Survey of Energy Harvesting Technologies for Wireless Sensor Networks," in IEEE Access, Vol. 9, 2021.

[29] F. Amin, A. Rashid, K. Salabat, and A. Muhammad, "An Overview of Medium Access Control and Radio Duty Cycling Protocols for Internet of Things," Electronics 11, No. 23, 2022.

[30] A. A. Lata and M. Kang, "A Review on Broadcasting Protocols for Duty-Cycled Wireless Sensor Networks," 2019 Eleventh International Conference on Ubiquitous and Future Networks (ICUFN), pp. 649-652, 2019.

[31] B. Chiara, A. Conti, D. Dardari, and R. Verdone, "An Overview on Wireless Sensor Networks Technology and Evolution," Sensors 9, No. 9, 2009.

[32] O. Khalaf and G. Abdulsahib, "Optimized Dynamic Storage of Data (ODSD) in IoT Based on Blockchain for Wireless Sensor Networks," in Peer-to-Peer Networking and Application, Vol. 14, No. 5, 2021.

[33] T. L. Le and M. H. Vo, "Lossless Data Compression Algorithm to Save Energy in Wireless Sensor Network," 2018 4th International Conference on Green Technology and Sustainable Development (GTSD), pp. 597-600, 2018.

[34] J. Sunil and E. M. Eyong. "An Energy Optimization in Wireless Sensor Networks by Using Genetic Algorithm," Telecommunication Systems 67, 2018.

[35] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues," in IEEE Communications Surveys & Tutorials, Vol. 21, No. 4, 2019.

[36] F. Engmann, F. A. Katsriku, J. Abdulai, K. S. Adu-Manu, and F. K. Banaseka. "Prolonging the Lifetime of Wireless Sensor Networks: A Review of Current Techniques," Wireless Communications and Mobile Computing, 2018.

[37] T. Bouguera, D. Jean-François, J. Chaillout, R. Jaouadi, and G. Andrieux, "Energy Consumption Model for Sensor Nodes Based on LoRa and LoRaWAN," Sensors 18, No. 7, 2018

[38] D. Crescini, F. Touati, and A. Galli, "Multiparametric Sensor Node for Environmental Monitoring Based on Energy Harvesting," Atmosphere 13, No. 2, 2022.

[39] H. Elayan, R. M. Shubair, and A. Kiourti, "Wireless Sensors for Medical Applications: Current Status and Future Challenges," 2017 11th European Conference on Antennas and Propagation (EUCAP), 2017.

[40] D. Kandris, C. Nakas, D. Vomvas, and G. Koulouras, "Applications of Wireless Sensor Networks: An Up-To-Date Survey," in Applied system innovation, Vol. 3, No. 1, 2020.

[41] D. Imededdin, A. Salih, and H. Medkour, "Design and Implementation of Low Power Consumption Wireless Sensor Node," Telkomnika (Telecommunication Computing Electronics and Control), Vol. 17, No. 6, 2019.

[42] S. Chelbi, H. Dhahri, and R. Bouaziz, "Node Placement Optimization Using Particle Swarm Optimization and Iterated Local Search Algorithm in Wireless Sensor Networks," in International Journal of Communication Systems, Vol. 34, No. 9, 2021.

[43] K. Cengiz and T. Dag, "Energy Aware Multi-Hop Routing Protocol for WSNs," in IEEE Access, Vol. 6, pp. 2622-2633, 2018.

[44] L. K. Ketshabetswe, A. M. Zungeru, M. Mangwala, and B. Sigweni. "Communication Protocols for Wireless Sensor Networks: A Survey and Comparison," Heliyon 5, No. 5, 2019.

[45] M. Khalil, A. Khalid, F. U. Khan and A. Shabbir, "A Review of Routing Protocol Selection for Wireless Sensor Networks in Smart Cities," 2018 24th Asia-Pacific Conference on Communications (APCC), pp. 610-615, 2018.

[46] M. Burunkaya and T. Pars, "A Smart Meter Design and Implementation Using Zigbee Based Wireless Sensor Network in Smart Grid," 2017 4th International Conference on Electrical and Electronic Engineering (ICEEE), pp. 158-162, 2017.

[47] B. K. Maharjan, U. Witkowski, and R. Zandian, "Tree Network Based on Bluetooth 4.0 for Wireless Sensor Network Applications," 2014 6th European Embedded Design in Education and Research Conference (EDERC), pp. 172-176, 2014.

[48] F. Ertam, I. F. Kilincer, O. Yaman, and A. Sengur, "A New IoT Application for Dynamic WiFi based Wireless Sensor Network," 2020 International Conference on Electrical Engineering (ICEE), pp. 1-4, 2020.

[49] A. Wixted, P. Kinnaird, H. Larijani, A. Tait, A. Ahmadinia, and N. Strachan, "Evaluation of LoRa and LoRaWAN for Wireless Sensor Networks," 2016 IEEE SENSORS, pp. 1-3, 2016.

[50] K. Kaur, P. Kaur, and E. Singh, "Wireless Sensor Network: Architecture, Design Issues and Applications," in International Journal of Scientific Engineering and Research (IJSER), Vol. 2, No. 11, 2014.

[51] K. Guleria and A. K. Verma, "Comprehensive Review for Energy Efficient Hierarchical Routing Protocols on Wireless Sensor Networks," Wireless Networks 25, 2019.

[52] H. Lu, J. Li, and M. Guizani, "Secure and Efficient Data Transmission for Cluster-Based Wireless Sensor Networks," in IEEE transactions on parallel and distributed systems, Vol. 25, No. 3, pp. 750-761, 2013.

[53] J. Sumathi and R. L. Velusamy, "A Review on Distributed Cluster Based Routing Approaches in Mobile Wireless Sensor Networks," in Journal of Ambient Intelligence and Humanized Computing, Vol. 12, No. 1, pp. 835-849, 2021.

[54] Z. Manap, B. M. Ali, Chee Kyun Ng, Nor Kamariah Noordin, and Aduwati Sali, "A Review on Hierarchical Routing Protocols for Wireless Sensor Networks." Wireless personal communications 72, 2013.

[55] A. Kanavalli, D. Sserubiri, P. D. Shenoy, K. R. Venugopal, and L. M. Patnaik, "A Flat Routing Protocol for Sensor Networks," In 2009 Proceeding of International Conference on Methods and Models in Computer Science (ICM2CS), pp. 1-5. IEEE, 2009.

[56] S. Mansour, N. Nasser, L. Karim, and A. Ali, "Wireless Sensor Network-based air quality monitoring system, " In Proceedings of the 2014 International Conference on Computing Networking and Communications (ICNC), pp. 545–550, 2014.

[57] A. Faustine, N. Mvuma, J. Mongi, C. Gabriel, J. Tenge, and B. Kucel, "Wireless Sensor Networks for Water Quality Monitoring and Control Within Lake Victoria Basin: Prototype Development," Wireless Sensor Network 6, No. 12, 2014.

[58] G. Paolini, M. Guermandi, D. Masotti, F. Benassi, L. Benini, and A. Costanzo, "RF-Powered Low-Energy Sensor Nodes for Predictive Maintenance in Electromagnetically Harsh Industrial Environments," Sensors 21, No. 2, 2021.

[59] T. Jabeen, I. Jabeen, H. Ashraf, Z. Jhanjhi, A. Yassine, and M. Hossain, "An Intelligent Healthcare System using IoT in Wireless Sensor Network," Sensors 23, No. 11, 2023.

[60] M. Rahu, S. Karim, R. Shams, A. Soomro, and A. Chandio, "Wireless Sensor Networks-Based Smart Agriculture: Sensing Technologies, Application and Future Directions," Sukkur IBA Journal of Emerging Technologies 5, No. 2, 2022.

[61] N. Suri, M. Tortonesi, J. Michaelis, P. Budulas, G. Benincasa, S. Russell, C. Stefanelli, and R. Winkler, "Analyzing the Applicability of Internet of Things to the Battlefield Environment," In 2016 international conference on military communications and information systems (ICMCIS), pp. 1-8. IEEE, 2016.

[62] M. P. Đurišić, Z. Tafa, G. Dimić, and V. Milutinović, "A Survey of Military Applications of Wireless Sensor Networks," in 2012 Mediterranean conference on embedded computing (MECO), pp. 196-199, 2012.

[63] M. Barnela and Suresh. Kumar. "Digital Modulation Schemes Employed in Wireless Communication: A Literature Review," International Journal of Wired and Wireless Communications 2, No. 2, 2014.

[64] Z. Chen, L. Wu, and P. Chen, "Efficient Modulation and Demodulation Methods for Multi-Carrier Communication," IET Communications 10, No. 5, 2016.

[65] K. Mahender, T. A. Kumar, and K. S. Ramesh, "Analysis of Multipath Channel Fading Techniques in Wireless Communication Systems," in AIP Conference Proceedings, Vol. 1952, No. 1, 2018.

[66] S.W. Roberts, "Control Chart Tests Based on Geometric Moving Averages," Technometrics 1, pp. 239–250, 1959.

[67] M. Zhang, J. Guo, X. Li, and R. Jin, "Data-Driven Anomaly Detection Approach for Time-Series Streaming Data," Sensors 20, No. 19, 2020.

[68] M. Zhang, X. Li, and L. Wang, "An Adaptive Outlier Detection and Processing Approach Towards Time Series Sensor Data," IEEE Access, Vol. 7, 2019.

[69] T. Mahmood, N. Balakrishnan, and M. Xie, "The Generalized Linear Model-Based Exponentially Weighted Moving Average and Cumulative Sum Charts for the Monitoring of High-Quality Processes," Applied Stochastic Models in Business and Industry 37, No. 4, 2021.

[70] J. Viinikka and H. Debar, "Monitoring IDS Background Noise Using EWMA Control Charts and Alert Information," In Recent Advances in Intrusion Detection: 7th International Symposium, pp. 166-187, 2004.

[71] B. Jiang, Y. Liu, H. Liu, Z. Ren, Y. Wang, and Y. Bao, "An Enhanced EWMA for Alert Reduction and Situation Awareness in Industrial Control Networks," IEEE 18th International Conference on Automation Science and Engineering (CASE), 2022.

[72] I. D. Mienye and Y. Sun, "A Survey of Ensemble Learning: Concepts, Algorithms, Applications, and Prospects," IEEE Access, Vol. 10, 2022.

[73] T. K. Ho, "Random Decision Forests," International Conference on Document Analysis and Recognition, Vol. 1, pp. 278-282, 1995.

[74] L. Breiman, "Bagging Predictors," Machine learning 24, pp. 123-140, 1996.

[75] R. Immonen and T. Hämäläinen, "Tiny Machine Learning for Resource-Constrained Microcontrollers," Journal of Sensors, Vol. 2022, No. 1, p. 7437023, 2022.

[76] A. Elhanashi, P. Dini, S. Saponara, and Q. Zheng, "Advancements in TinyML: Applications, Limitations, and Impact on IoT Devices," Electronics, Vol. 13, No. 17, p. 3562, 2024.

[77] E. Abanelli, G. Tagliavini, and L. Benini, "Optimizing Random Forest-Based Inference on RISC-V MCUs at the Extreme Edge," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 41, No. 11, pp. 4516–4526, 2022.

[78] Y. Abadade, A. Temouden, H. Bamoumen, N. Benamar, Y. Chtouki, and A. S. Hafid, "A Comprehensive Survey on TinyML," IEEE Access, vol. 11, pp. 96892–96922, 2023.

Declaration

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been duly acknowledged.

Declared by:

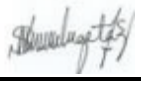
Name: Alemayehu Ebissa

Signature: _____

Date: _____

Confirmed by advisor:

Name: Dr. Mulugeta Libsie

Signature: 

Date: 19 Oct. 2024