



# Long-Range Wide Area Network Intrusion Detection at the Edge

Gonçalo Esteves <sup>1</sup>, Filipe Fidalgo <sup>1</sup>, Nuno Cruz <sup>1,2,\*</sup> and José Simão <sup>1,3</sup>

<sup>1</sup> Future Internet Technologies—FIT, Instituto Superior de Engenharia de Lisboa, Instituto Politécnico de Lisboa, 1500-335 Lisbon, Portugal; gesteves@deetc.isel.ipl.pt (G.E.); a42109@alunos.isel.pt (F.F.); jose.simao@isel.pt (J.S.)

<sup>2</sup> LASIGE, Faculdade de Ciências, Universidade de Lisboa, 1749-016 Lisboa, Portugal

<sup>3</sup> INESC-ID Lisboa, 1000-029 Lisboa, Portugal

\* Correspondence: ncruz@isel.ipl.pt

**Abstract:** Internet of Things (IoT) devices are ubiquitous in various applications, such as smart homes, asset and people tracking, and city management systems. However, their deployment in adverse conditions, including unstable internet connectivity and power sources, present new cybersecurity challenges through new attack vectors. The LoRaWAN protocol, with its open and distributed network architecture, has gained prominence as a leading LPWAN solution, presenting novel security challenges. This paper proposes the implementation of machine learning algorithms, specifically the K-Nearest Neighbours (KNN) algorithm, within an Intrusion Detection System (IDS) for LoRaWAN networks. Through behavioural analysis based on previously observed packet patterns, the system can detect potential intrusions that may disrupt critical tracking services. Initial simulated packet classification attained over 90% accuracy. By integrating the Suricata IDS and extending it through a custom toolset, sophisticated rule sets are incorporated to generate confidence metrics to classify packets as either presenting an abnormal or normal behaviour. The current work uses third-party multi-vendor sensor data obtained in the city of Lisbon for training and validating the models. The results show the efficacy of the proposed technique in evaluating received packets, logging relevant parameters in the database, and accurately identifying intrusions or expected device behaviours. We considered two use cases for evaluating our work: one with a more traditional approach where the devices and network are static, and another where we assume that both the devices and the network are mobile; for example, when we need to report data back from sensors on a rail infrastructure to a mobile LoRaWAN gateway onboard a train.



**Citation:** Esteves, G.; Fidalgo, F.; Cruz, N.; Simão, J. Long-Range Wide Area Network Intrusion Detection at the Edge. *IoT* **2024**, *5*, 871–900.  
<https://doi.org/10.3390/iot5040040>

Academic Editor: Amiya Nayak

Received: 11 October 2024

Revised: 18 November 2024

Accepted: 30 November 2024

Published: 4 December 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

**Keywords:** IoT; LoRaWAN; intrusion detection system; machine learning

## 1. Introduction

The Internet of Things (IoT) has experienced rapid growth and widespread adoption across various sectors, from smart homes and cities to industrial systems and healthcare. This proliferation of connected devices has revolutionized data collection, analysis, and automation, enabling unprecedented levels of efficiency and insight. However, the sheer scale and diversity of IoT deployments have introduced new challenges, particularly in terms of communication protocols and network security.

Among the various communication protocols developed for IoT [1], Long-Range Wide Area Network (LoRaWAN) has emerged as a leading solution, particularly for applications requiring long-range low-power communication. LoRaWAN's ability to transmit small amounts of data over long distances while minimizing power consumption has made it an attractive choice for a wide range of IoT applications, from agricultural monitoring to urban infrastructure management [2].

The widespread adoption of LoRaWAN, driven by its decentralized and open network architecture, has introduced significant security challenges that demand urgent attention. These networks often carry sensitive data and control critical infrastructure, making their

security paramount. LoRaWAN's unique characteristics, such as low-bandwidth communication, geographically dispersed nodes, and large-scale deployments, necessitate specialized security measures that work within these constraints. However, the open nature of the network and the resource limitations of IoT devices expose it to various vulnerabilities, including man-in-the-middle attacks, network flooding, traffic analysis, replay attacks, and physical tampering with devices in remote or unsecured locations. Addressing these issues is essential to safeguard the integrity and reliability of LoRaWAN deployments.

Traditional intrusion detection systems (IDS) are often ill-suited for LoRaWAN environments due to the unique characteristics of these networks [3,4]. The low data rates, sporadic transmission patterns, and large-scale deployments typical of LoRaWAN networks require specialized approaches to security monitoring. Moreover, the potential for network disruption in critical applications, such as emergency response systems or industrial control networks, makes timely and accurate intrusion detection crucial.

Security challenges in LoRaWAN are especially serious in situations with poor connectivity or mobile deployments, like search and rescue operations in remote areas or monitoring infrastructure in isolated locations. In these cases, relying on centralized, cloud-based security solutions can cause delays or fail due to lack of network access. To ensure the safety and reliability of these networks, strong and decentralized intrusion detection systems that work well at the network edge are needed. Solving these problems is important for protecting LoRaWAN systems, building trust in IoT technology, and encouraging its wider use in critical applications.

Although IoT security has been extensively studied, the specific challenges posed by LoRaWAN networks have received less attention. Most existing research focuses on traditional security measures such as encryption, authentication protocols, and centralized monitoring systems. While effective in some contexts, these approaches often fail to address the unique constraints of LoRaWAN environments. Some researchers have explored the use of machine learning for intrusion detection in IoT networks, demonstrating promising results in identifying anomalies and potential threats [5–7]. However, applying these techniques to the distinct traffic patterns and security challenges of LoRaWAN networks remains limited. Moreover, many current solutions depend on centralized processing, which can introduce latency and create single points of failure.

Recent studies highlight the potential of edge computing to improve IoT security by reducing latency, enhancing privacy, and increasing resilience to network disruptions. Edge-based solutions are particularly well-suited to LoRaWAN, given its constraints and decentralized nature. However, the integration of edge computing with machine learning-based intrusion detection systems for LoRaWAN presents a significant research gap. Addressing this issue could lead to more robust, efficient, and scalable security frameworks tailored to LoRaWAN's specific needs.

To address these critical gaps in LoRaWAN security, we propose a novel intrusion detection system that leverages edge computing and machine learning techniques. Our approach combines the strengths of decentralized processing with adaptive anomaly detection to create a robust, efficient, and scalable security solution for LoRaWAN networks.

At the core of our system is a machine learning model based on the K-Nearest Neighbours (KNN) algorithm, chosen for its effectiveness in classifying network traffic patterns and its relatively low computational requirements. This model is trained on real-world LoRaWAN network data, enabling it to accurately distinguish between normal traffic and potential intrusions based on subtle variations in packet metadata and signal characteristics.

We integrate this machine learning model with the Suricata Intrusion Detection System (IDS), a powerful open-source tool that we have extended and customized for LoRaWAN environments. This integration allows us to combine traditional signature-based detection with our behaviour-based anomaly detection, providing a multi-layered approach to identifying security threats.

Our system is designed to both operate at the network core and edge, either near the network server or implemented directly on or near LoRaWAN gateways. We consider that the edge-based deployment offers several key advantages:

1. Reduced latency in threat detection and response;
2. Continued operation even in scenarios with limited or intermittent backhaul connectivity;
3. Enhanced privacy by processing sensitive data locally;
4. Improved scalability through distributed processing.

The edge scenario that considers a distributed gateway approach offers potential benefits for low-latency LoRaWAN applications. Two illustrative use cases demonstrate the advantages of this approach:

1. **Mountain search and rescue:** In emergency scenarios, response teams could employ edge gateways in their vehicles to track hikers' LoRaWAN GPS tracker signals in areas with limited cellular and static LoRaWAN coverage. This allows for real-time location tracking and rapid response in critical situations.
2. **Railway safety monitoring:** A train could be equipped with an edge gateway to receive information directly from sensors deployed on slopes adjacent to the tracks. These sensors could detect potential hazards such as rockfalls or ground sliding. By processing these data in real-time on the moving train, the system could provide immediate alerts to the train operator, enabling rapid response to potential dangers without relying on centralized infrastructure or experiencing delays in data transmission.

These use cases highlight the flexibility and responsiveness of the distributed gateway architecture, particularly in scenarios where low latency and local processing are crucial for safety and efficiency. The ability to process data at the edge, whether on a mobile search and rescue vehicle or a moving train, demonstrates the potential of this architecture to enhance real-time decision-making and response in various critical applications.

Our system uses adaptive learning to adjust to changing network conditions and evolving threats, making it suitable for dynamic environments like mobile or temporary deployments in emergency scenarios. It also introduces a method to decrypt payloads at the edge for thorough security analysis without breaking end-to-end encryption or adding significant overhead. These features together create an efficient and flexible intrusion detection system for LoRaWAN, addressing key gaps in current solutions and improving the security of IoT deployments.

The primary objectives of this research are to develop a robust, edge-based intrusion detection system specifically tailored for LoRaWAN networks. This study aims to leverage machine learning techniques, particularly the K-Nearest Neighbours algorithm, for effective anomaly detection in LoRaWAN traffic patterns. Additionally, this research seeks to design an adaptive system that is capable of operating effectively in both static and dynamic network environments, including mobile and temporary deployments. Finally, this study intends to validate the proposed system using real-world LoRaWAN network data, ensuring its applicability and effectiveness in practical scenarios.

The key contributions of this work to the field of IoT and LoRaWAN security are significant and multifaceted. We present a novel integration of edge computing and machine learning, combining edge computing capabilities with machine learning-based intrusion detection specifically optimized for LoRaWAN networks. This approach addresses latency and connectivity challenges inherent in centralized security solutions.

Our research develops and implements an adaptive KNN model for LoRaWAN traffic analysis, effectively classifying network traffic to distinguish between normal patterns and potential intrusions. The model is trained on real-world data, enhancing its accuracy and relevance in operational environments.

We extend the open-source Suricata IDS to work effectively with LoRaWAN protocol specifics, creating a powerful hybrid system that combines signature-based and anomaly-based detection methods. Our decentralized security architecture enables autonomous

operation at the network edge, improving resilience and reducing dependence on constant backhaul connectivity.

This study provides comprehensive evaluation insights from testing in both stationary and mobile scenarios, offering valuable data on edge-based intrusion detection performance in varying network conditions.

Importantly, our work offers real-world validation by utilizing data from an operational LoRaWAN network in Lisbon, providing empirical evidence of the system's effectiveness in real-world conditions. This represents a significant step beyond simulated or small-scale testbed evaluations.

Our research methodology encompasses several key components and phases. We began with data collection and analysis, utilizing real-world LoRaWAN network data from a gateway operated by the Lisbon Polytechnic Engineering School (ISEL), Portugal. This dataset, comprising over 500,000 messages from thousands of devices, was thoroughly analysed to understand typical traffic patterns and characteristics.

The system was evaluated in both centralized and edge scenarios, including gateway mobility, using metrics such as detection accuracy, false positive rates, and processing latency. Adaptive learning mechanisms were implemented, enabling periodic model retraining and dynamic threshold adjustments to handle changing network conditions and evolving threats. By combining theoretical analysis with practical implementation and real-world testing on actual LoRaWAN data, the methodology ensures both academic rigour and practical relevance, offering insights applicable to real-world IoT security challenges.

The remainder of this work is organized as follows: Section 2 reviews related work on IoT security, LoRaWAN security challenges, and machine learning in network intrusion detection; Section 3 covers system implementation, including data preprocessing, model training, and Suricata customization; Section 4 analyses the detection model performance; and, Section 5 concludes with key contributions and future research directions.

## 2. Related Work

The rapid proliferation of Internet of Things (IoT) devices and networks has introduced new security challenges, particularly in low-power wide-area network (LPWAN) protocols like LoRaWAN. As these networks become increasingly prevalent in various applications, from smart cities to industrial systems, ensuring robust security measures becomes paramount. This section reviews relevant literature on intrusion detection systems (IDS) for IoT environments, with a particular focus on LoRaWAN networks and the application of machine learning techniques.

We begin by examining general approaches to IDS in IoT contexts, followed by LoRaWAN-specific security challenges and solutions. We then explore various machine learning algorithms applied to network security, particularly in the realm of intrusion detection. Finally, we consider the emerging paradigm of edge computing and its implications for IoT security. Through this review, we aim to identify current trends, challenges, and gaps in the research landscape, providing context for our proposed KNN-based intrusion detection system for LoRaWAN networks.

### 2.1. Intrusion Detection Systems for IoT Networks

The proliferation of Internet of Things (IoT) devices has introduced new security challenges to network infrastructures, necessitating specialized Intrusion Detection Systems (IDS). Unlike traditional networks, IoT ecosystems are characterized by heterogeneous devices, constrained resources, and diverse communication protocols, requiring unique approaches to security [6].

Conventional IDS solutions are often inadequate in addressing the complexities of IoT networks, particularly in detecting unauthorized access attempts and novel attack vectors [8]. This inadequacy has driven research towards more sophisticated detection mechanisms, often leveraging artificial intelligence and machine learning techniques to

provide deeper insights into system behaviour and enhance the detection of novel attack patterns.

Recent advancements in IDS for IoT have explored various machine learning algorithms. Ref. [9] presents a comprehensive comparison of Logistic Regression (LR), Support Vector Machines (SVMs), Decision Trees (DTs), and Artificial Neural Networks (ANNs) for implementing anomaly-based IDS, utilizing public datasets such as IoTID20 [10] and BoT-IoT [11] to evaluate their efficacy in detecting common IoT network attacks.

The application of K-Nearest Neighbours (KNN) and Decision Tree-based classification for IoT intrusion detection has been explored in [7], though challenges in real-time detection due to data normalization requirements have been noted. Ref. [12] further demonstrates the effectiveness of KNN in network intrusion detection for IoT environments, specifically as a Network-based Intrusion Detection System (NIDS).

An innovative approach by Mohammed Baz [13] introduces SEHIDS, a self-evolving host-based intrusion detection system employing artificial neural networks within individual IoT nodes. While promising, this approach raises concerns about feasibility in highly resource-limited devices.

The distinction between Host-based Intrusion Detection Systems (HIDS) and Network-based Intrusion Detection Systems (NIDS) in IoT contexts is discussed in [14], which incorporates LoRaWAN-specific radio data (RSSI-SNR) into the detection process. However, this approach does not account for gateway location, which could be particularly relevant in mobile edge computing scenarios.

As the IoT landscape continues to evolve, so too must the approaches to securing these networks. The integration of machine learning algorithms into IDS frameworks represents a promising direction in adapting security measures to the dynamic nature of IoT threats [15]. These intelligent systems offer the potential for more accurate threat detection, reduced false positives, and improved overall network resilience in the face of increasingly sophisticated cyber attacks.

Future research should focus on developing lightweight, distributed IDS that can operate effectively across heterogeneous IoT devices and protocols, as well as exploring the integration of context-aware and collaborative IDS approaches to further enhance the security posture of IoT networks.

## 2.2. LoRaWAN-Specific Security Challenges and Solutions

As LoRaWAN emerges as a prominent LPWAN technology for IoT applications, it brings unique security challenges that require specialized attention. The open and distributed nature of LoRaWAN networks introduces vulnerabilities that differ from traditional network architectures.

In [16], the authors provide a comprehensive overview of various attacks specific to LoRaWAN networks, including Man-in-the-Middle (MITM) attacks, network flooding attacks, network traffic analysis, physical attacks, Radio Frequency (RF) jamming attacks, and self-replay attacks. While software-based defences can address many of these threats, physical attacks that compromise gateway access remain a significant concern, highlighting the need for multi-layered security approaches.

The vulnerabilities of the LoRaWAN protocol to replay, jamming, wormhole, and flipping attacks have been identified and analysed in [17]. This research particularly focuses on jamming attacks as a denial-of-service vector against IoT devices. The study employs advanced algorithms, including Kullback–Leibler divergence and Hamming distance measures, to achieve high accuracy in traffic analysis and attack detection.

Addressing the need for LoRaWAN-specific security solutions, [5] explores the application of artificial intelligence algorithms for analysing LoRaWAN device behaviours. Their approach combines two analysis techniques: an initial k-means algorithm to group devices with similar behavioural patterns, followed by Decision Tree (DT) and Long Short-Term Memory (LSTM) models to predict expected behaviour. This dual-layer approach enhances the ability to detect anomalies and potential security breaches in LoRaWAN networks.



The unique characteristics of LoRaWAN, such as its use of unlicensed spectrum and long-range capabilities, necessitate security measures that can operate effectively within the constraints of low-power devices and limited bandwidth.

The integration of machine learning techniques with LoRaWAN-specific security measures shows promise in enhancing threat detection capabilities. By leveraging the unique characteristics of LoRaWAN traffic patterns and device behaviours, ML-based intrusion detection systems can potentially identify subtle anomalies that might indicate security breaches or attempts at unauthorized access.

### *2.3. Machine Learning Algorithms for Network Security*

The application of machine learning algorithms in network security, particularly for intrusion detection systems, has gained significant traction in recent years. These algorithms offer the potential for more accurate, adaptive, and efficient threat detection compared to traditional rule-based systems.

The K-Nearest Neighbours (KNN) algorithm has emerged as a popular choice for network intrusion detection due to its simplicity and effectiveness. Ref. [18] demonstrates the use of KNN for intrusion detection in conventional network environments, testing it on network traffic data from an air force local area network. The algorithm's ability to classify data points based on their proximity to known samples makes it particularly suitable for identifying anomalous network behaviour.

Building upon this, [19] provides a detailed implementation of KNN for data classification in network security contexts. The study highlights KNN's method of calculating the probability of a point being close to other points, effectively quantifying relationships between data points. This characteristic is particularly useful in distinguishing between normal and anomalous network traffic patterns.

Decision trees represent another powerful machine learning approach for network security. Ref. [20] implements a rules and decision tree-based intrusion detection system for IoT environments. This research emphasizes the increasing sophistication of cyberattacks, particularly those targeting critical infrastructure and sensitive information systems. The hierarchical nature of decision trees allows for interpretable decision-making processes, which can be crucial in understanding and explaining detected threats.

Sparse Logistic Regression (SLR) offers a unique approach to network intrusion detection, as demonstrated in [21]. By incorporating lasso regularization into the logistic regression framework, SLR introduces sparsity into the model. This feature enables automatic selection of the most relevant features from the data, allowing key indicators that distinguish between normal and abnormal network activities to be identified efficiently.

The integration of machine learning algorithms with existing IDS frameworks has also shown promising results. Ref. [22] combines the Suricata IDS with KNN to classify IoT botnet attacks on resource-constrained devices like Raspberry Pi. This approach demonstrates the potential for implementing sophisticated ML-based intrusion detection, even in environments with limited computational resources.

However, it is important to note that the effectiveness of machine learning algorithms in network security can vary depending on the specific context and type of network. For instance, while [18] focuses on traditional network traffic, the application of these algorithms to IoT and specifically LoRaWAN networks presents unique challenges and opportunities. The distinct characteristics of LoRaWAN traffic, including low data rates and long transmission intervals, require adaptations to conventional ML approaches.

As network architectures become more complex and diverse, particularly with the advent of edge computing and 5G networks, the role of machine learning in network security is likely to become even more critical. Future research should focus on developing ML algorithms that can operate effectively across heterogeneous network environments while maintaining high accuracy and low false positive rates.

#### 2.4. Edge Computing in IoT Security

The emergence of edge computing has introduced new paradigms in IoT security, particularly in the context of intrusion detection systems. By bringing computation and data storage closer to the source of data generation, edge computing offers potential benefits in terms of reduced latency, improved privacy, and enhanced security for IoT networks.

In [23], the authors provide an overview of attack vectors in edge computing environments and propose the use of machine learning systems for threat detection. However, it is important to note that this research primarily focuses on Multi-Edge Access Computing (MEC) in the context of NB-IoT and 5G networks, which differs from the LoRaWAN context of our study.

The implementation of IDS at the edge presents both opportunities and challenges. On the one hand, edge-based IDS can potentially detect and respond to threats more quickly due to their proximity to the data source. This is particularly relevant for time-sensitive IoT applications where rapid threat detection is crucial [14].

However, edge devices often have limited computational resources compared to centralized cloud infrastructure. This constraint necessitates the development of lightweight, efficient IDS solutions that can operate effectively within these limitations. Ref. [8] highlights the importance of considering resource constraints when designing security solutions for IoT environments, which is equally applicable to edge computing scenarios.

In the context of LoRaWAN networks, edge computing can play a significant role in enhancing security. By implementing IDS functionalities at the gateway level, it becomes possible to analyse traffic patterns and detect anomalies closer to the source, potentially before malicious traffic reaches the core network. Ref. [14] touches on this concept by considering radio data (RSSI-SNR) in their IDS approach, though they do not fully explore the implications of gateway mobility in edge scenarios.

The integration of machine learning algorithms with edge-based IDS presents promising opportunities for enhancing IoT security. For instance, the KNN algorithm, which has shown effectiveness in IoT intrusion detection, as demonstrated by [12], could potentially be adapted for edge deployment. However, careful consideration must be given to the algorithm's computational requirements and the need for frequent model updates in dynamic IoT environments.

Furthermore, edge computing introduces new challenges in terms of maintaining consistent security policies across distributed nodes. The decentralized nature of edge architectures requires innovative approaches to key management, device authentication, and access control that can operate effectively in environments with intermittent connectivity [24].

As edge computing continues to evolve, there is a growing need for research into adaptive, scalable IDS solutions that can leverage the benefits of edge architecture while addressing its unique challenges. Future work should focus on developing edge-native security frameworks that can provide robust protection for IoT networks without compromising the performance benefits that edge computing offers [25].

#### 2.5. Research Gaps

Our review of the current literature reveals several key areas for future research in IoT and LoRaWAN security. There is a need for better integration of LoRaWAN characteristics with edge computing capabilities in IDS solutions, as well as the development of lightweight, adaptive ML algorithms for edge-based intrusion detection. Work should also focus on creating context-aware and real-time adaptive IDS solutions to address the dynamic threat landscape.

Holistic security approaches combining traditional IDS with physical layer security techniques remain underexplored. The field would benefit from standardized benchmarks and datasets specifically for LoRaWAN security research, as well as investigations into collaborative and distributed IDS architectures for large-scale deployments.

Notably, current research largely assumes static gateway positions, overlooking scenarios with mobile LoRaWAN gateways. There is a significant gap in developing security

solutions for mobile applications, such as gateways onboard moving vehicles or in other dynamic environments.

Addressing these gaps is crucial for developing robust security solutions that can effectively protect the growing ecosystem of IoT and LoRaWAN networks, particularly as they become increasingly integral to critical infrastructure and services.

### 3. Implementation

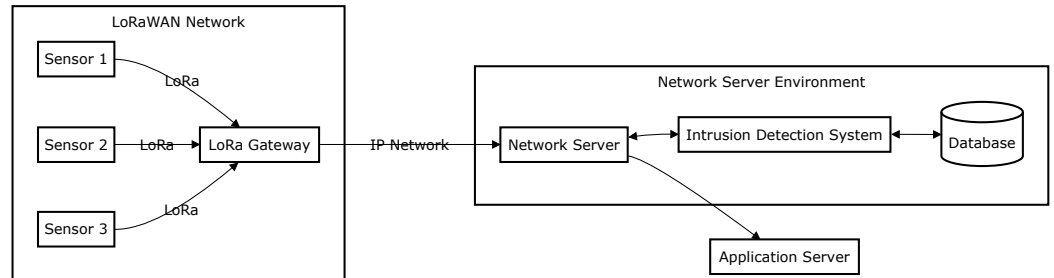
This section details the design and implementation of our proposed Intrusion Detection System (IDS) for LoRaWAN networks. We begin by outlining the system architecture, followed by a comprehensive analysis of real-world LoRaWAN data that informed our approach. We then describe the core components of our system, including the open-source IDS, database, and machine learning algorithms. The implementation process is explained in detail, covering aspects such as data preprocessing, model training, and packet classification. Finally, we discuss the adaptation of our system for edge computing environments, highlighting the benefits and challenges of this approach. Throughout this section, we emphasize the rationale behind our design choices and the techniques employed to enhance the security and efficiency of LoRaWAN networks.

#### 3.1. System Architecture

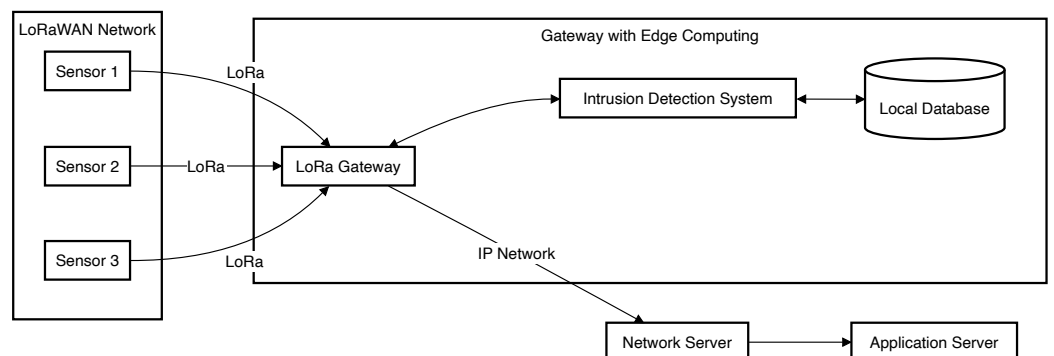
Our Intrusion Detection System (IDS) for LoRaWAN networks is designed to operate at the IoT device level, utilizing behavioural analysis through machine learning models. The system architecture accommodates two potential deployment configurations:

1. IDS installed on/or near each LoRaWAN gateway;
2. IDS installed at the Network Server (NS), receiving packets from all gateways.

Figures 1 and 2 illustrate these two architectural approaches:



**Figure 1.** Architecture for IDS in the NS.



**Figure 2.** Architecture for IDS in or near each LoRaWAN gateway.



In the centralized network server deployment (Figure 1), multiple gateways may forward identical messages at varying times, introducing an additional analysis dimension not addressed in this study. Conversely, the distributed gateway deployment (Figure 2) reduces latency associated with LoRaWAN Network Server (NS) communication but requires each gateway to possess packet decryption capabilities.

The data flow in our system begins with sensor data transmitted via the LoRaWAN protocol to the gateways. These gateways encapsulate device payloads into IP packets using Semtech's packet forwarder format for relaying to the network server. Our IDS analyses these packet streams post-gateway, regardless of the chosen deployment configuration.

### 3.2. Data Analysis

To ensure the realism and effectiveness of our machine learning models, we utilized real-world device data in our implementation. This section details our data source, preprocessing methods, and key findings from our analysis.

We obtained sample data from a LoRaWAN gateway operated by Instituto Superior de Engenharia de Lisboa (ISEL), located in Lisbon's Amoreiras Towers. This gateway, connected to The Things Network (TTN), receives an average of 500,000 messages monthly from thousands of devices.

The dataset was preprocessed using the Pandas Profiling library [26] to generate a comprehensive analysis report. This preprocessing was performed on Google Colab to facilitate collaboration.

Our examined sample is from June 2020, during the peak of COVID-19 spread in Portugal. The dataset consists of 509,042 messages sent from 2876 devices to the network server. From the various message parameters, we selected the most relevant for distinguishing devices.

#### 3.2.1. Key Parameters Analysis

The selection of Received Signal Strength Indicator (RSSI), Signal-to-Noise Ratio (SNR), Spreading Factor (SF), and payload length as key parameters for analysis is based on their availability and relevance in LoRaWAN networks, particularly in the context of encrypted payloads. These parameters provide quantifiable metrics for network behaviour analysis without compromising payload confidentiality. RSSI and SNR offer quantitative measures of signal quality and strength, enabling the detection of anomalies in transmission patterns. SF, a core parameter in LoRa modulation, allows for the identification of changes in device configuration. Payload length serves as a potential indicator of abnormal behaviour, as deviations from expected message sizes may signify unauthorized transmissions. This parameter set enables the development of an intrusion detection system that maintains the integrity of LoRaWAN payload encryption while facilitating effective security monitoring through analysis of readily available metadata.

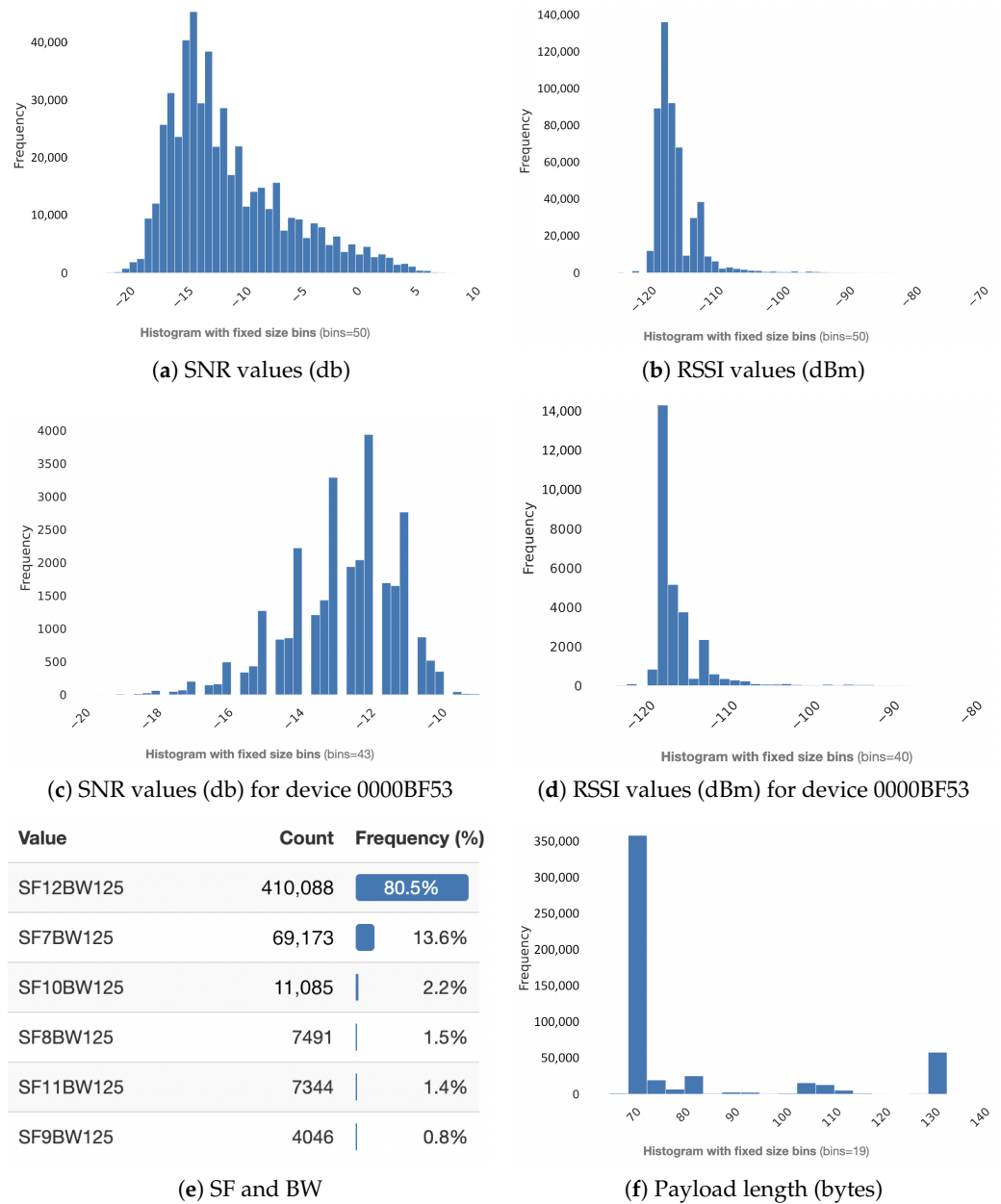
#### Signal-to-Noise Ratio (SNR) and Received Signal Strength Indicator (RSSI)

Figure 3a–d illustrate the trends in SNR and RSSI across the samples and illustrate the details for one of the devices. Our analysis revealed that the signal-to-noise ratio clusters around  $-15$  dB, while the RSSI values are notably concentrated around  $-118$  dBm. These findings suggest that the majority of devices are in fixed geographical positions.

#### Spreading Factor (SF), Bandwidth (BW), and Payload Length

Figure 3e shows the distribution of SF and BW values in the sample. Our analysis shows that the most common configuration uses a Spreading Factor (SF) of 12 and Bandwidth (BW) of 125, which is related to the fact that the Adaptive Data Rate (ADR) bit is active in frame control. Indicating that it intends to make ADR, the network server is able to change the SF, BW, or transmission power of the device in order to optimise binary throughput, propagation time, and power consumption. Figure 3f indicates the distribution

of payload lengths in the sample. The payload lengths predominantly cluster around 70 bytes.



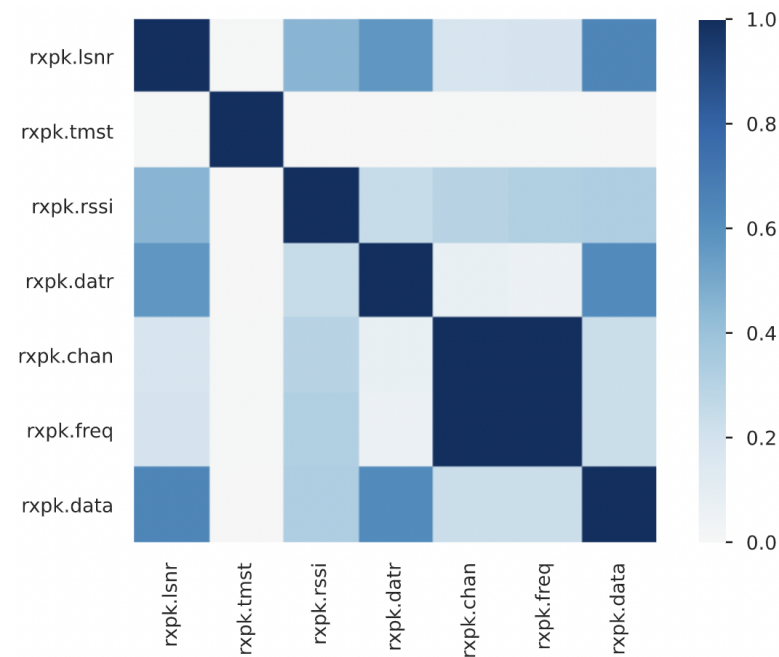
**Figure 3.** Dataset characteristics.

### 3.2.2. Parameter Correlations

To assess the utility of various parameters for our learning model, we conducted an inter-parameter correlation analysis. Figure 4 illustrates the derived Phik correlations, providing insight into the relationships between categorical and ordinal variables within our dataset. This correlation method is designed to work with mixed data types. The advantages include working with both continuous and categorical variables, handling non-linear relationships, and being less sensitive to outliers [27].

The correlation analysis revealed several key findings that inform our understanding of the data structure and its implications for our model. As anticipated, the parameters demonstrated the highest correlations with themselves, serving as a baseline for comparison. As expected, we observed a strong inter-correlation between the “chan” (channel) and

“freq” (frequency) parameters, which is consistent with their representation of essentially the same information in different formats.



**Figure 4.** Phik correlation between different variables of the dataset.

Of particular interest was the behaviour of the payload size parameter “data”. This parameter exhibited elevated correlations with several signal metrics, including “snr” (signal-to-noise ratio), “datr” (data rate), and “rssi” (received signal strength indicator). This relationship aligns well with the fundamental principles of LoRaWAN functionality, where payload size can influence and be influenced by various transmission characteristics.

Interestingly, the timestamp parameter “tmst” stood out for its lack of correlation with any other parameter when considered in isolation. This independence suggests that temporal factors in our dataset may not have a strong direct relationship with other measured variables, potentially indicating consistent network behaviour over time or the need for more complex temporal analysis (e.g., periodicity) in future work.

These correlation findings provide valuable insights into the interdependencies within our dataset and will guide our feature selection and model development processes in subsequent stages of our research.

### 3.3. System Components

Our implemented intrusion detection system comprises three main components: an open-source Intrusion Detection System (IDS), a database, and Python 3.8 or higher software for executing machine learning algorithms. This section details each component and its role within the system.

#### 3.3.1. Open-Source IDS: Suricata

In the selection of an Intrusion Detection System (IDS) for our LoRaWAN security implementation, we conducted a comprehensive evaluation of various open-source options, including Snort and Zeek [28,29]. After careful consideration, Suricata emerged as the optimal choice due to its unique combination of features and capabilities that align closely with the requirements of our research.

Suricata’s robust packet-level analysis capabilities enable detailed examination of network traffic through signature-based intrusion detection (SIDS).

The architecture of Suricata offers several advantages that are particularly relevant to our research objectives. Its multi-threading support, designed to fully utilize multi-core

processors, enables efficient processing of high-volume traffic—a feature especially beneficial for scalable LoRaWAN deployments. Furthermore, Suricata’s extensibility through Lua scripting facilitates seamless integration with databases and supplementary Python scripts, allowing us to tailor the IDS specifically to LoRaWAN protocol characteristics. This feature aligns well with our research approach, which aims to enhance detection capabilities through advanced analytical techniques. This support streamlines the incorporation of our custom machine learning models, potentially improving the overall effectiveness of our IDS.

Performance considerations also played a significant role in our selection process. Benchmark studies have demonstrated Suricata’s competitive performance in terms of throughput and detection capabilities when compared to alternatives such as Snort. This performance edge is critical for maintaining the efficiency of our intrusion detection system in real-world LoRaWAN environments.

Additionally, the active open-source community and comprehensive documentation surrounding Suricata provide valuable resources for ongoing development and troubleshooting. This ecosystem support is crucial for the long-term viability and improvement of our research implementation.

While other IDS solutions like Snort offer comparable features, Suricata’s unique combination of performance, extensibility, and community support made it the most suitable choice for our LoRaWAN-focused implementation. The ability to seamlessly extend Suricata’s functionality through Lua scripts and integrate it with our custom Python-based machine learning models was a decisive factor, aligning closely with our research methodology and objectives.

### 3.3.2. Database: CrateDB

For the database component of our system, we evaluated several time-series databases including InfluxDB, TimescaleDB, and CrateDB. We ultimately selected CrateDB, as it best aligns with the specific requirements of our research [30]. CrateDB’s SQL-based architecture was a key factor in our decision, as it employs familiar SQL syntax for query operations and data management while providing built-in time-series optimization. This combination offers an ideal balance of usability and performance for analysing temporal patterns in LoRaWAN traffic compared to the custom query languages required by alternatives like InfluxDB.

The scalability of CrateDB was another important consideration, given the potential high message volume characteristic of LoRaWAN networks. Its distributed architecture allows for seamless scaling across nodes, which is crucial for handling large-scale deployments that may involve thousands of devices. While TimescaleDB offers similar scaling capabilities, CrateDB’s design provides additional flexibility in managing large data volumes, which is essential for processing and analysing the extensive network traffic generated in our study.

Furthermore, CrateDB’s optimization for time-series data proved particularly relevant to our research needs. This feature aligns well with the temporal nature of network traffic data, enabling more effective storage and retrieval of time-stamped information. The database’s support for real-time querying and analysis is essential for our IDS implementation, enabling rapid detection of potential security threats. Additionally, CrateDB’s ability to handle both structured and semi-structured data accommodates the varying metadata parameters in LoRaWAN packets while maintaining query performance.

These features collectively provide a robust foundation for managing and analysing the complex data generated in our LoRaWAN security research, supporting both our current analytical needs and potential future expansions of our work.

### 3.3.3. Machine Learning: Python Programs

For the implementation of our machine learning algorithms, we utilized Python due to its robust ecosystem of libraries and frameworks designed for data analysis and model

development. This choice allowed us to leverage powerful tools throughout our research process, from initial data processing to final model deployment. Our data preprocessing phase employed custom Python scripts to extract and prepare relevant parameters from the LoRaWAN packets. These scripts were designed to efficiently handle the unique characteristics of LoRaWAN data, ensuring that the extracted features were optimally formatted for subsequent analysis and model training. For model training and evaluation, we primarily utilized the scikit-learn library [31], a widely-recognized tool in the machine learning community. Scikit-learn provided us with a comprehensive set of algorithms (e.g., KNN) and evaluation metrics, enabling us to develop and assess our intrusion detection models. This approach allowed for systematic comparison of different modelling techniques and hyperparameter configurations. The final component of our Python-based machine learning implementation focused on real-time classification. We developed Python programs capable of performing rapid classification of incoming packets based on our trained models. This real-time processing capability is crucial for the practical application of our intrusion detection system in live LoRaWAN environments. By utilizing Python throughout our machine learning pipeline, from data preprocessing to real-time classification, we were able to create a cohesive and efficient system for LoRaWAN intrusion detection. This integrated approach facilitated seamless data flow between different stages of our analysis and allowed for flexible iteration and improvement of our models throughout the research process.

#### 3.3.4. K-Nearest Neighbours (KNN) Algorithm

For our intrusion detection system, we selected the K-Nearest Neighbours (KNN) algorithm due to its demonstrated effectiveness in classification tasks and its particular suitability for the characteristics of LoRaWAN network data. KNN is a non-parametric, instance-based learning algorithm that has shown robust performance in various network security applications [12,18]. Non-parametric models offer advantages in security threat detection through their ability to identify attack patterns. Unlike parametric models, which rely on fixed parameters and distributions, implementations like K-nearest neighbours and decision trees adapt dynamically to new attack signatures [32].

The selection of KNN was based on several key factors that align well with the requirements of our research. Firstly, the simplicity and interpretability of KNN's approach facilitate easy implementation and interpretation, which is crucial for the ongoing maintenance and updating of the IDS. This transparency in decision-making is particularly valuable in security applications where understanding the rationale behind classifications is often as important as the classifications themselves.

KNN's effectiveness with smaller datasets was another significant consideration. LoRaWAN networks typically generate less-voluminous data (per device) compared to traditional IP networks, and KNN's ability to perform well with limited training data makes it particularly suitable for our use case. This characteristic allows us to develop effective models, even in scenarios where extensive historical data may not be available.

The algorithm's natural ability to handle multi-class problems was also a key factor in our decision. This capability is beneficial for distinguishing between various types of network behaviours and potential intrusions, allowing for more nuanced classification beyond simple binary (normal/intrusion) categorization.

Furthermore, KNN's status as a lazy learning algorithm, requiring minimal training time, aligns well with the dynamic nature of network security. This low training overhead allows for quick adaptation to new patterns in network traffic; a crucial feature in the rapidly evolving landscape of cybersecurity threats.

Lastly, KNN's suitability for continuous learning was a significant advantage. The algorithm can easily incorporate new instances, facilitating ongoing learning and adaptation to evolving network conditions. This feature is particularly valuable in the context of LoRaWAN networks, where new devices and usage patterns may emerge over time.

While we considered other machine learning algorithms such as Support Vector Machines (SVMs) and random forests, KNN’s balance of simplicity, effectiveness, and adaptability made it the most suitable choice for our LoRaWAN intrusion detection system. Our implementation builds upon the work of Liao and Vemuri [18], who demonstrated KNN’s effectiveness in network intrusion detection, and extends it to the specific context of LoRaWAN networks. This approach allows us to leverage proven methodologies while adapting them to the unique characteristics of LoRaWAN environments.

### 3.3.5. System Integration

Figure 5 illustrates how these components interact within our system:

1. Suricata analyses incoming network packets and triggers the preprocessing module.
2. The preprocessing module extracts relevant parameters from LoRaWAN packets relayed by the packet forwarder.
3. Python scripts insert the processed data into the CrateDB database.
4. Machine learning models, implemented in Python, analyse the data stored in CrateDB to detect potential intrusions.
5. The results of the analysis are fed back into Suricata for alert generation and further action.

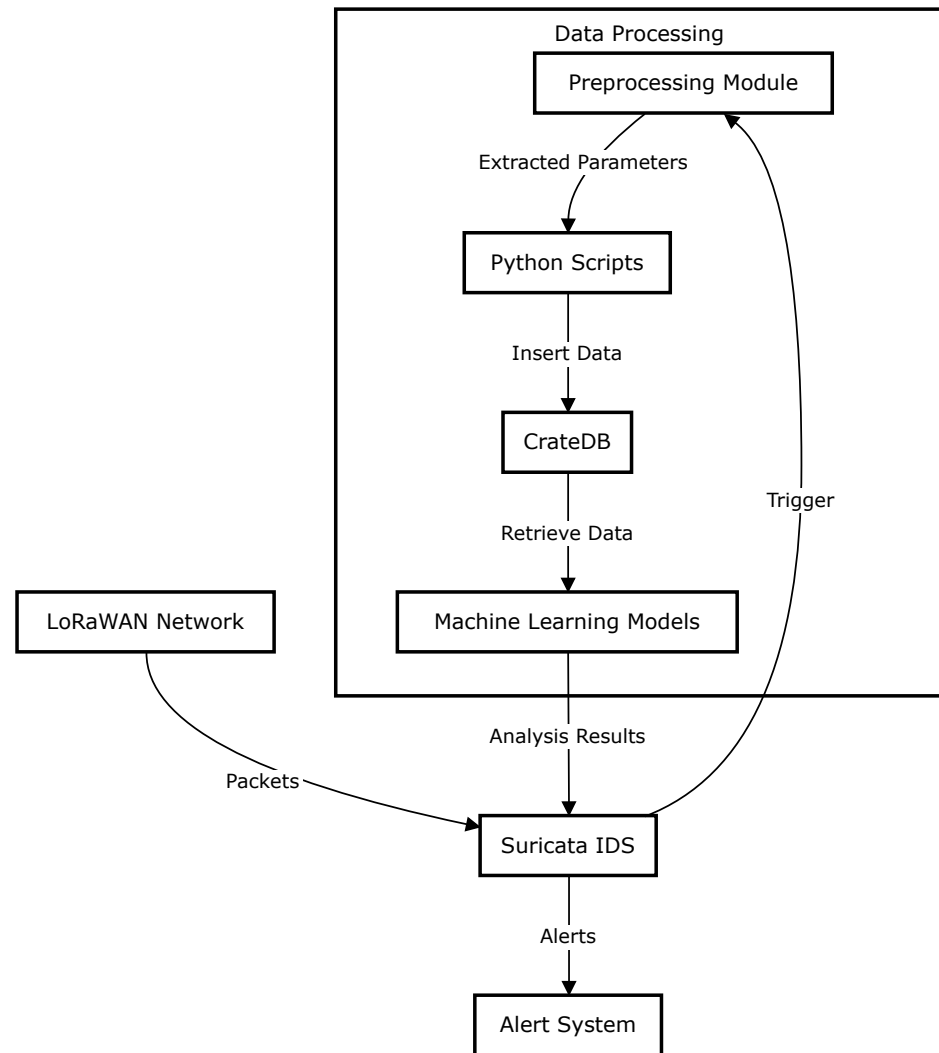


Figure 5. Functional architecture.



This integrated approach allows for efficient, real-time intrusion detection in LoRaWAN networks, combining the strengths of signature-based detection (through Suricata) with the adaptability of machine learning-based anomaly detection.

### 3.4. Evaluation Setup

This section details the step-by-step process of implementing our Intrusion Detection System (IDS) for LoRaWAN networks. We describe the configuration of Suricata, the packet classification process, and the integration of our machine learning model.

#### 3.4.1. Suricata Configuration

Our Suricata configuration was tailored to detect and analyse LoRaWAN packets efficiently. The configuration process involved three key components: signature implementation, Lua script integration, and Python integration.

For signature implementation, we developed a custom signature specifically designed to identify LoRa gateway packets destined for the network server. This signature serves as the initial filter for incoming network traffic, ensuring that only relevant LoRaWAN packets are processed further.

Upon detection of a matching signature, our system triggers a Lua script for preprocessing. This script is responsible for initial parsing and preparation of the received packet contents, extracting relevant features for subsequent analysis.

The final stage of our configuration utilizes Python for database integration. We chose Python due to its comprehensive libraries, which facilitate efficient insertion of preprocessed parameters into our CrateDB database.

Our current implementation processes packets individually as they traverse from the gateway to the network server. While bulk processing might seem more efficient in other contexts, the relatively low data rates characteristic of LoRaWAN networks make this packet-level analysis approach feasible. To optimize performance and minimize intrusion detection latency, we leverage Suricata's multi-core CPU processing capabilities.

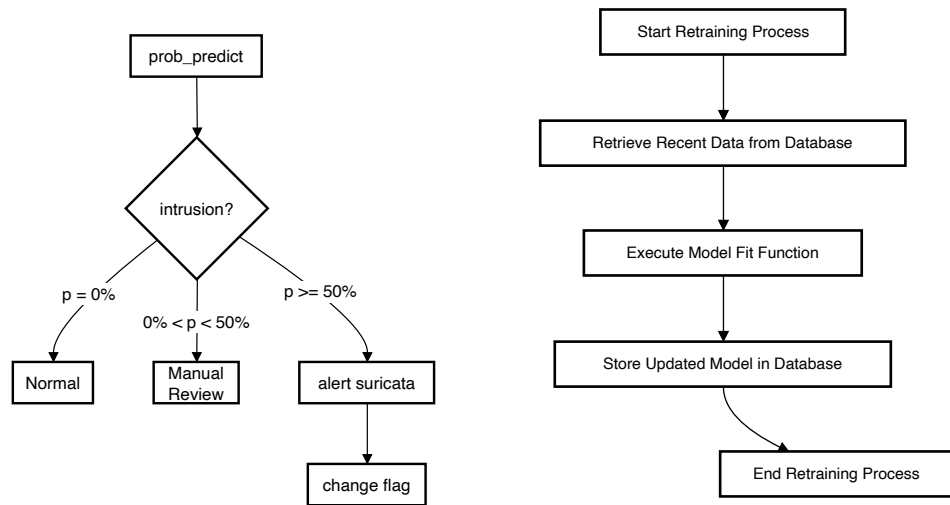
#### 3.4.2. Packet Classification Process

The packet classification process in our system follows a structured approach designed to efficiently identify potential intrusions in LoRaWAN traffic. This process is initiated when Suricata flags a new packet, triggering our custom classification algorithm.

Upon detection of a new packet, the system retrieves the corresponding pre-trained model from our database, based on the packet's DevAddr. This device-specific model, trained on historical data, forms the basis for our classification.

The core of our classification process is the probability prediction phase, where our `prob_predict` function evaluates the likelihood of the packet being an intrusion. This function applies the retrieved model to the packet's features, generating a probability score.

The classification decision is then made based on this probability score. Packets with a probability exceeding 50% are classified as intrusions, prompting a Suricata alert and an update to the database flag. For probabilities between 0% and 50%, the system marks the packet for manual review, allowing for human verification of borderline cases. Packets with a 0% probability are classified as normal traffic and require no further action. This tiered approach to classification allows for a nuanced treatment of network traffic, balancing automated detection with human oversight for ambiguous cases. Figure 6a provides a visual representation of this classification process, illustrating the decision flow from initial packet detection to final classification.



(a) Predict flowchart

(b) Flowchart of the system's retraining

Figure 6. Packet classification flowcharts.

### 3.4.3. Model Retraining Process

To maintain the effectiveness of our intrusion detection model over time, we have implemented a retraining process that allows for periodic updates. This process is designed to adapt the model to potential changes in network behaviour and emerging threat patterns. The current implementation focuses on core functionality of model retraining and updating, as illustrated in Figure 6b.

The retraining process begins with data retrieval from our database, specifically accessing the most recent data for the device in question. This step ensures that the model is updated with the latest available information, potentially capturing new patterns or behaviours that have emerged since the previous training session.

Following data retrieval, the model retraining phase is executed using the `fit` function. This process updates the model's parameters based on the new data, enabling it to adapt to changes in the network's behaviour or threat landscape. Once retraining is complete, the newly trained model is stored in the database, replacing the previous version to ensure all future predictions utilize the most current model.

Currently, the initiation of the retraining process is manual, providing flexibility in determining when updates are necessary. This approach allows system administrators to schedule retraining based on various factors such as significant changes in network infrastructure, the emergence of new threat types, or as part of regular maintenance routines.

While our current implementation does not include an automated confidence scoring system, it establishes a foundation for maintaining model relevance over time. The straightforward nature of this approach facilitates quick updates when needed, ensuring the intrusion detection system can adapt to evolving network conditions.

Future enhancements may include the implementation of automated triggers for retraining based on performance metrics or time intervals. However, the current manual initiation allows for careful control and observation of the retraining process, providing valuable insights into the model's behaviour and the network's changing characteristics over time.

This retraining methodology aims to balance the need for model adaptability with the requirement for controlled and interpretable updates in the context of network security.

### 3.4.4. Initial Training Phase

Our implemented model requires an initial training phase on normal network traffic and behaviour. This phase establishes baseline metrics, enabling the system to detect deviations that may constitute intrusions. Only after this initial learning period can the model accurately distinguish between expected and suspicious activity.

### 3.4.5. Relevant Parameters

Table 1 lists the key parameters we use for intrusion detection:

**Table 1.** Relevant parameters for intrusion detection.

Parameter	Description
tmst	Timestamp
latitude	Gateway latitude
longitude	Gateway longitude
chan	Communication channel
bw	Bandwidth
sf	Spreading factor
rssi	Received signal power
snr	Signal-to-noise ratio
chanlenpayload	Payload length in bytes
payload	Payload content
flag	Intrusion indication

The “flag” parameter is initially set by Suricata to classify the packet as non-intrusive. After analysis by our algorithm, this flag may be updated to indicate an intrusion if detected.

This implementation process ensures a comprehensive and adaptive approach to intrusion detection in LoRaWAN networks, combining the strengths of signature-based anomaly detection with machine learning-based anomaly detection.

## 3.5. Edge Computing Implementation

This section details the adaptation of our Intrusion Detection System (IDS) for edge computing environments, highlighting the benefits and challenges of this approach in the context of LoRaWAN networks.

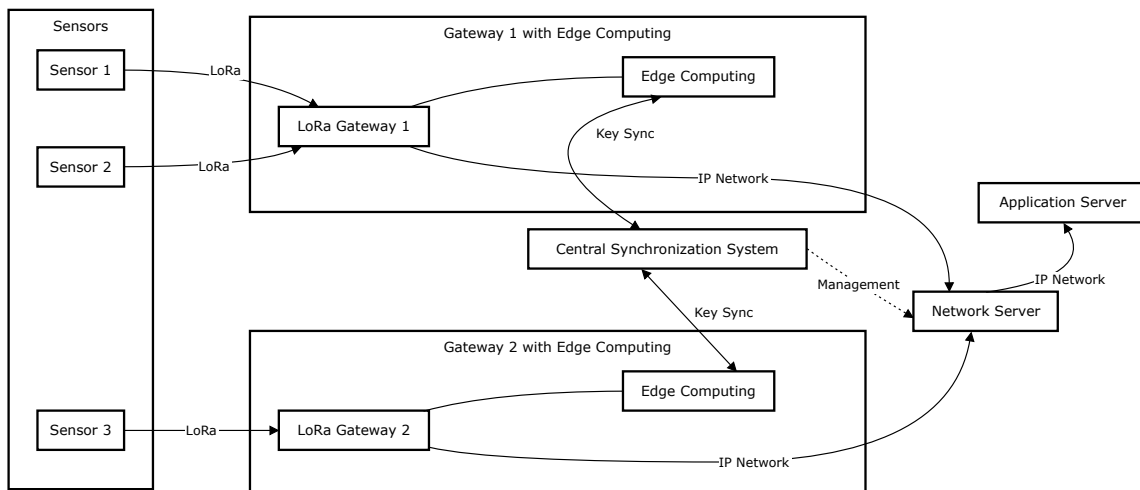
### 3.5.1. Edge Computing in LoRaWAN Context

Edge computing adopts a decentralized data processing approach, strategically deploying computing resources near data sources. In our LoRaWAN context, this localizes processing to the ingress via LoRaWAN gateways. This approach offers several advantages, including reduced latency, improved privacy, enhanced security, and increased real-time throughput capabilities. The traditional LoRaWAN architecture does not support this model, requiring data to be reported back to a central application server. To achieve this, we had to change the architecture to allow a minimal network server service at/near the LoRaWAN gateways. This allows for intrusion detection at the payload level. However, our work is focused more on intrusion detection considering an encrypted payload; therefore, we have not included the decrypted payload as one of the parameters for the IDS, which also allows for a more direct comparison between the two scenarios.

### 3.5.2. Architecture for Edge Computing

Implementing our IDS in an edge environment necessitates incorporating autonomous detection within each gateway. Figure 7 illustrates this architecture:

In this architecture, sensors connect to gateways via LoRa radio technology. The data are then received by an edge computing platform, ideally housed within the LoRaWAN gateway, where it they are decrypted and processed. A central system connects all edge computing platforms within the network, facilitating coordination and data sharing.



**Figure 7.** Schematic of LoRaWAN connection between sensors and the network using an edge computing environment.

### 3.5.3. Role of the Central Synchronization Server

The central server plays a crucial role in maintaining synchronization among the various edge computing nodes. It enables the sharing of vital information such as device EUIs, network IDs, network keys, and application keys among all gateways and edge computing platforms. Additionally, it facilitates Over-the-Air Activation (OTAA) authentication, a fundamental component of the LoRaWAN protocol, ensuring all edge computing platforms and gateways are aligned with necessary network parameters.

### 3.5.4. Benefits and Challenges of Edge Computing in IDS

Implementing our IDS in an edge computing environment offers several advantages. It reduces latency by processing data at the edge, eliminating delays associated with transmitting the data to a central server for analysis. The approach improves reliability, as each gateway functions independently, ensuring continued operation even if connectivity to the central synchronization server is lost. It also enhances security by allowing sensitive data to be processed locally, reducing the risk of interception during transmission. Furthermore, edge computing enables real-time analysis of LoRaWAN traffic, allowing for immediate detection and response to potential intrusions.

However, edge computing also presents challenges. Edge devices may have limited computational resources compared to centralized servers, necessitating careful optimization of our machine learning models and algorithms. Ensuring all edge devices have up-to-date intrusion detection models can be challenging, requiring efficient model distribution mechanisms. Coordinating intrusion detection across multiple edge devices demands careful system design. Moreover, edge devices themselves may become targets for attacks, requiring additional security measures.

### 3.5.5. Implementation Considerations

When implementing our IDS in an edge computing environment, we consider several key factors. We focus on resource optimization, ensuring our machine learning models and algorithms run efficiently on resource-constrained edge devices. We implement a distributed learning approach where edge devices can contribute to model improvement without centralizing all data. Secure communication is prioritized, with all communication between edge devices and the central synchronization server being encrypted and authenticated. Lastly, we implement fallback mechanisms to ensure continued operation in case of connectivity issues with the central synchronization server.

By addressing these considerations, we can leverage the benefits of edge computing to create a more efficient, responsive, and robust intrusion detection system for LoRaWAN

networks. This approach allows us to harness the power of localized processing while maintaining the coordination and oversight benefits of a centralized system.

### 3.6. Assumptions and Limitations

The development and implementation of our Intrusion Detection System (IDS) for LoRaWAN networks is predicated on several key assumptions and is subject to certain limitations. Understanding these factors is required for interpreting the system's capabilities and results, as well as for identifying areas for future research and improvement.

Our IDS implementation assumes the availability of a period of normal network traffic for initial model training. This assumption is fundamental to establishing baseline behaviour patterns against which potential intrusions can be detected. The importance of this initial training period cannot be overstated, as it directly impacts the system's ability to accurately distinguish between normal and anomalous network behaviour. However, this assumption also presents a challenge in dynamic network environments where 'normal' may be a moving target.

Another significant assumption is the relative stability of the network topology. While LoRaWAN networks are generally more static compared to some other wireless network types, any substantial changes in network structure may necessitate model retraining. This assumption highlights the need for adaptive learning mechanisms in future iterations of the system to accommodate network evolution without compromising detection accuracy.

The system also operates under the assumption of consistent device behaviour over time. This premise allows for the establishment of behavioural profiles for legitimate devices, with significant deviations from these profiles serving as potential indicators of intrusion. However, this assumption may not hold in all cases, particularly for devices with legitimately variable behaviour patterns, potentially leading to false positives.

In terms of limitations, our current implementation faces challenges in detecting highly sophisticated, adaptive attacks that gradually alter their behaviour to mimic legitimate traffic patterns. This limitation is common to many machine learning-based security systems and underscores the ongoing arms race between security measures and evolving attack methodologies. Addressing this limitation will require the development of more nuanced, possibly unsupervised learning techniques that can detect subtle, long-term behavioural shifts.

False positives represent another significant limitation of our system, as with many IDS implementations. While our approach strives to minimize false alarms, the system may still flag legitimate but unusual network behaviour as potentially malicious. This limitation highlights the delicate balance between sensitivity and specificity in intrusion detection, as well as the need for continuous model tuning and refinement based on operational feedback.

The system's reliance on metadata and traffic pattern analysis, while preserving data privacy, also limits its effectiveness against attacks that do not significantly alter these observable characteristics. This constraint is particularly relevant in the context of encrypted communications, where payload analysis is not feasible. Future research could explore methods of incorporating encrypted payload analysis without compromising data confidentiality.

Scalability presents another challenge, particularly in large-scale LoRaWAN deployments. The volume of data and number of devices in such deployments may strain computational resources, especially in edge computing implementations. This limitation points to the need for more efficient algorithms and possibly hierarchical or distributed processing approaches to manage large-scale networks effectively.

Lastly, the effectiveness of our machine learning models is inherently dependent on the quality and quantity of available historical data. In scenarios with limited or non-representative historical data, the system's detection accuracy may be compromised. This limitation underscores the importance of data collection and curation in the deployment of machine learning-based security systems.

Looking ahead, our research will focus on addressing these limitations and expanding the system's capabilities. Key areas for future work include the development of more adaptive models that are capable of automatically adjusting to gradual changes in network behaviour, the implementation of advanced techniques for false positive reduction, the exploration of privacy-preserving methods for encrypted payload analysis, the investigation of distributed learning techniques for improved scalability, and the development of strategies for effective operation with limited historical data.

In conclusion, while our IDS for LoRaWAN networks represents a significant step forward in securing these increasingly important networks, it is crucial to understand its underlying assumptions and current limitations. This understanding not only allows for more effective deployment and interpretation of results but also guides the direction of future research and development efforts in this critical area of network security.

#### 4. Results and Discussion

This section presents the experimental results of our proposed Intrusion Detection System (IDS) for LoRaWAN networks, implemented in two distinct environments: a conventional network-connected server and an edge computing platform. We evaluate the performance of our machine learning-based approach using standard metrics including accuracy, recall, and F1 score. The analysis is divided into two main parts: first, we examine the results from the centralized server environment, then we explore the performance in the edge computing scenario. For each environment, we provide both quantitative assessments and qualitative interpretations of the data. Finally, we offer a comparative analysis of the two approaches, highlighting their respective strengths and limitations in the context of LoRaWAN security.

##### 4.1. Experimental Setup

Our experiments were conducted in two distinct environments to evaluate the performance and scalability of the proposed intrusion detection system:

1. **Conventional network-connected server:** This setup represents a traditional centralized architecture where all packet analysis is performed on a single server connected to the LoRaWAN network.
2. **Edge computing platform:** In this configuration, the IDS is implemented directly on LoRaWAN gateways, enabling localized packet analysis and intrusion detection at the network edge.

The IDS employs a KNN algorithm for anomaly detection, trained on features extracted from LoRaWAN packet metadata, including signal strength (RSSI), signal-to-noise ratio (SNR), and packet size. For the edge computing environment, we also incorporated gateway GPS location data to account for its mobility.

We have established two datasets for each scenario, where each dataset is comprised of all the messages for a single device. Each dataset was split into a training set (70% of the messages) and a test set (30% of the messages).

On the test dataset, we introduced variations in packet size, SNR, and RSSI. These variations were intentionally introduced to represent potential anomalies, as fixed sensors typically exhibit limited fluctuations under normal conditions. Significant variations in SNR and RSSI could indicate changes in device position, while alterations in bandwidth and packet size might suggest modified messages or device configurations.

##### 4.2. Performance Metrics

To comprehensively evaluate the effectiveness of our intrusion detection system, we employed three standard performance metrics that are widely used in machine learning and security applications:



1. **Accuracy:** This metric represents the overall correctness of the model and is defined as follows:

$$Accuracy = \frac{NumberOfCorrectlyClassifiedMessages}{TotalNumberOfMessages} \quad (1)$$

Accuracy provides a general measure of the model's performance but can be misleading in cases of class imbalance.

2. **Recall:** Also known as sensitivity or the true positive rate, recall measures the model's ability to correctly identify all positive instances (in our case, intrusions). It is calculated as follows:

$$Recall = \frac{NumberOfTruePositives}{NumberOfActualPositives} \quad (2)$$

Recall is particularly important in intrusion detection, as it quantifies the system's capability to detect all potential threats.

3. **F1 score:** This is the harmonic mean of precision and recall, providing a balanced measure of the model's performance:

$$F1Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (3)$$

where precision is defined as follows:

$$Precision = \frac{NumberOfTruePositives}{NumberOfPredictedPositives} \quad (4)$$

The F1 score is valuable when seeking a balance between precision and recall, which is crucial in intrusion detection to minimize both false positives and false negatives.

These metrics were chosen to provide a comprehensive view of our IDS performance. Accuracy gives an overall performance measure, recall ensures we are not missing potential intrusions, and the F1 score balances the trade-off between precision and recall. Together, they offer a robust evaluation of the system's effectiveness in both centralized and edge computing environments.

#### 4.3. Centralized Server Environment Results

This section presents the results of the IDS system when using the centralized server approach.

##### 4.3.1. Dataset Characteristics

For the centralized server environment, we used a subset of all messages from the dataset comprising LoRaWAN packets selected for a single device. The dataset was further divided as follows:

- Total messages: 590
- Training set: 413 messages (70%)
- Test set: 177 messages (30%)

##### 4.3.2. Model Performance

Our machine learning-based IDS demonstrated robust performance in the centralized server environment. The overall classification accuracy consistently exceeded 90%, indicating strong general performance. Specifically:

- Correctly classified non-intrusive messages: 556
- Accurately detected intrusions: 30
- Uncertain classifications: 4

##### 4.3.3. Quantitative Analysis

We calculated the performance metrics as follows:

$$Accuracy = \frac{556 + 30}{590} \approx 0.9932 \text{ or } 99.32\% \tag{5}$$

$$Recall = \frac{30}{30} = 1.0 \text{ or } 100\% \tag{6}$$

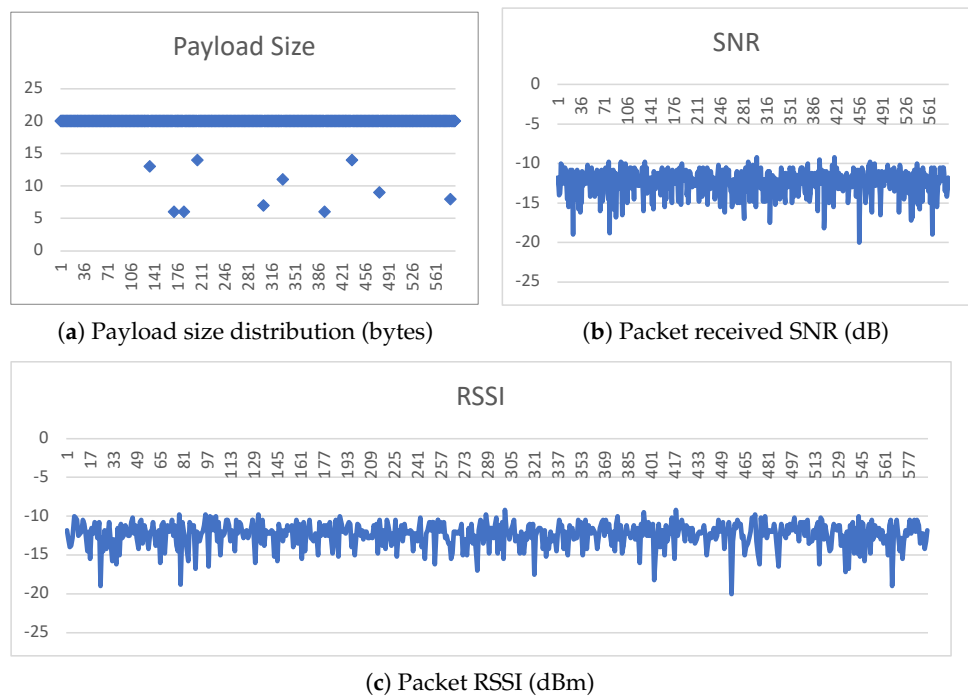
$$Precision = \frac{30}{30 + 3} \approx 0.9091 \text{ or } 90.91\% \tag{7}$$

$$F1Score = 2 \cdot \frac{0.9091 \cdot 1.0}{0.9091 + 1.0} \approx 0.9524 \text{ or } 95.24\% \tag{8}$$

These results indicate excellent performance, with perfect recall and high precision, resulting in a strong F1 score.

#### 4.3.4. Qualitative Analysis

Figure 8 illustrates the characteristics of the packets in our test dataset.

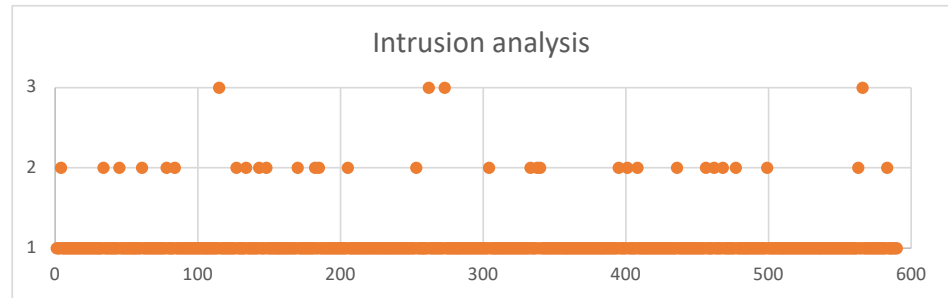


**Figure 8.** Characteristics of the packets in the test dataset for the centralized server scenario.

Figure 9 displays the classification results for each packet. In this figure, the classifications are labelled as follows:

- 1: Non-suspicious packet
- 2: Suspicious packet (potential intrusion)
- 3: Uncertain classification

The model demonstrates robust performance in distinguishing between normal and suspicious packets, even with the introduced variability in SNR and RSSI. This aligns with the high accuracy and F1 score calculated in our quantitative analysis. The results from the centralized server environment suggest that our IDS can effectively detect potential intrusions in a LoRaWAN network, with high accuracy and recall. The system’s ability to handle variations in packet characteristics while maintaining high performance is particularly noteworthy.



**Figure 9.** Intrusion detection results in the centralized server environment.

#### 4.4. Edge Computing Environment Results

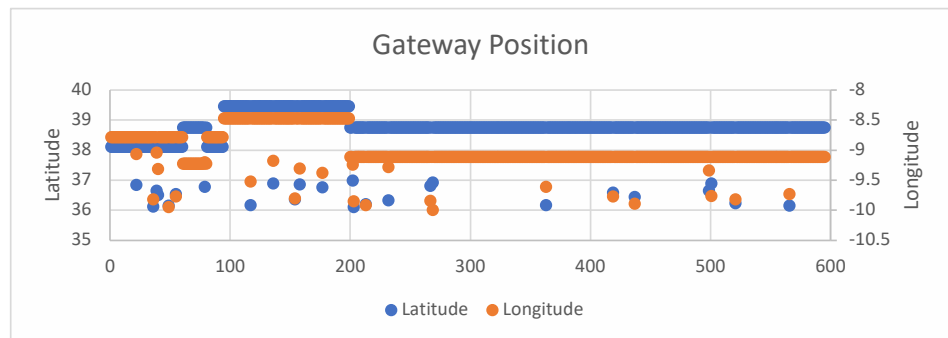
This section presents the results of the IDS system when used in the edge computing environment.

##### 4.4.1. Dataset Characteristics

In the edge computing scenario, we utilized a new subset of the dataset with the following characteristics:

- Total messages: 595
- Training set: 417 messages (70%)
- Test set: 178 messages (30%)

In the current scenario, we simulated gateway mobility to assess the system’s performance under dynamic conditions. This involved periodically changing the gateway’s location during data collection and analysis, as shown in Figure 10.



**Figure 10.** Locations of the gateway during the edge computing experiment.

##### 4.4.2. Model Performance

The IDS implemented in the edge computing environment demonstrated good performance, albeit with some challenges introduced by gateway mobility. The overall classification accuracy was approximately 90%. Specifically:

- Correctly classified non-intrusive messages: 513
- Accurately detected intrusions: 25
- Uncertain or misclassified: 57

##### 4.4.3. Quantitative Analysis

We calculated the performance metrics as follows:

$$Accuracy = \frac{513 + 25}{595} \approx 0.9042 \text{ or } 90.42\% \tag{9}$$

$$Recall = \frac{25}{25} = 1.0 \text{ or } 100\% \tag{10}$$

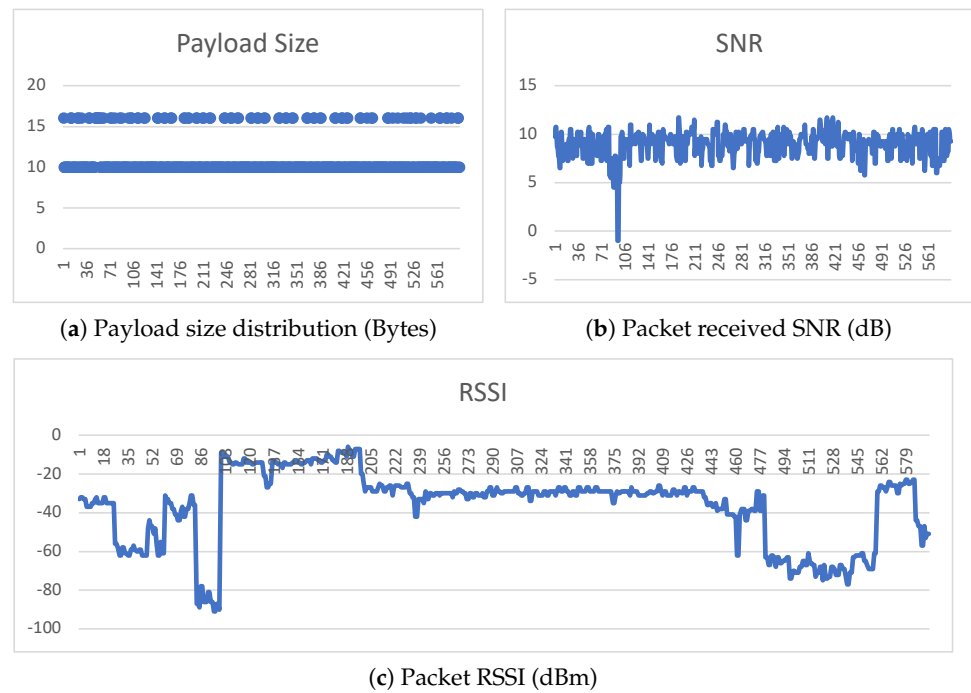
$$Precision = \frac{25}{29} \approx 0.8621 \text{ or } 86.21\% \tag{11}$$

$$F1 \text{ Score} = 2 \cdot \frac{0.8621 \cdot 1.0}{0.8621 + 1.0} \approx 0.9259 \text{ or } 92.59\% \tag{12}$$

These results indicate good performance, with perfect recall but lower precision compared to the centralized environment, resulting in a slightly lower F1 score.

#### 4.4.4. Qualitative Analysis

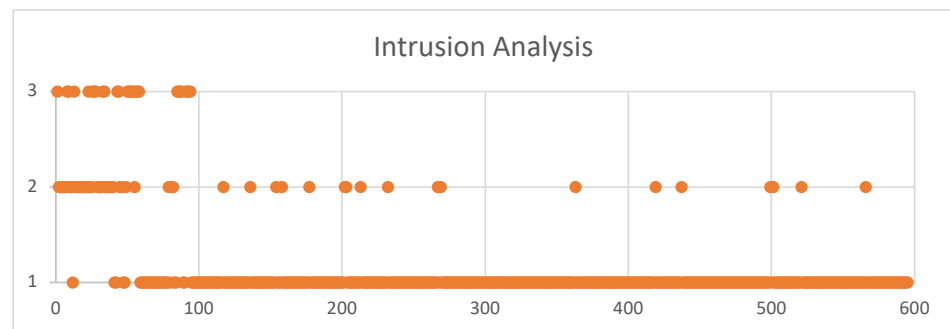
Figure 11 illustrates the characteristics of the packets in our edge computing test dataset.



**Figure 11.** Characteristics of the packets in the test dataset for the edge computing scenario.

This figure shows significant variations in RSSI and SNR due to the mobility of the edge computing environment and sensors. These rapid changes pose challenges for the model, as a sensor can experience a wider range of SNR and RSSI values compared to the static centralized scenario.

Figure 12 displays the classification results for each packet in the edge environment.



**Figure 12.** Intrusion detection results in the edge computing environment.

As in the centralized environment, the classifications are labelled as follows:

- 1: Non-suspicious packet
- 2: Suspicious packet (potential intrusion)
- 3: Uncertain classification

The model maintains good performance in distinguishing between normal and suspicious packets, but with an increased number of uncertain classifications compared to the centralized environment.

The changing gateway positions contribute to the variations in RSSI and SNR observed in Figure 11, presenting additional challenges for the IDS.

The results from the edge computing environment demonstrate that our IDS can effectively detect potential intrusions in a mobile LoRaWAN network, maintaining high recall but with reduced precision compared to the centralized setup. The system's performance in handling the increased variability introduced by mobility is promising, though it presents opportunities for further optimization.

#### 4.5. Comparative Analysis

To better understand the strengths and limitations of our IDS in different deployment scenarios, we compare the performance metrics between the centralized server and edge computing environments.

Table 2 summarizes the key performance metrics for both environments. Several important observations can be made:

1. **Accuracy:** The centralized server environment achieved higher overall accuracy (99.32% vs. 90.42%). This difference can be attributed to the more stable conditions in the centralized setup, where network characteristics remain relatively constant.
2. **Recall:** Both environments maintained perfect recall (100%), indicating that the IDS successfully identified all intrusions in both scenarios. This is a critical achievement for a security system, as it suggests no threats were missed.
3. **Precision:** The centralized server showed higher precision (90.91% vs. 86.21%), meaning it had fewer false positives. The lower precision in the edge environment can be attributed to the challenges posed by mobility and varying signal strengths.
4. **F1 score:** The F1 scores, while both high, favour the centralized environment (95.24% vs. 92.59%). This reflects the balance between the perfect recall in both scenarios and the higher precision in the centralized setup.

**Table 2.** Performance comparison between centralized and edge computing environments.

Metric	Centralized Server	Edge Computing
Accuracy	99.32%	90.42%
Recall	100%	100%
Precision	90.91%	86.21%
F1 score	95.24%	92.59%

The impact of mobility on model performance is evident in the edge computing results. The variations in RSSI and SNR due to changing gateway positions (as seen in Figure 11) created more challenging conditions for accurate classification. This resulted in a higher number of uncertain classifications and slightly lower precision.

However, it is important to note the trade-offs between these two approaches:

- **Latency:** The edge computing approach offers the potential for lower latency in threat detection, as analysis occurs closer to the data source.
- **Scalability:** Edge computing can provide better scalability for large LoRaWAN deployments by distributing the computational load.
- **Robustness:** The edge approach may offer greater robustness in scenarios with unreliable network connectivity to a central server.

- **Accuracy:** The centralized approach currently offers higher accuracy and precision, benefiting from a more stable environment and potentially more computational resources.

While the centralized server environment showed superior performance in terms of accuracy and precision, the edge computing approach demonstrated resilience in a more challenging, mobile scenario. The ability to maintain perfect recall and achieve a high F1 score in the edge environment is particularly noteworthy, suggesting that our IDS can effectively adapt to the dynamic conditions of mobile LoRaWAN deployments.

#### 4.6. Computational Requirement Analysis

To evaluate the practical applicability of our KNN-based IDS in dynamic conditions, we analysed its computational requirements based on our implementation data:

- **Memory requirements:** The KNN model's memory footprint consists primarily of training data storage, requiring approximately  $O(nd)$  of space [33], where  $n$  is the number of training samples (417 messages in our implementation) and  $d$  is the number of features (9 parameters per message, as detailed in Table 1).
- **Algorithmic complexity:** Our implementation has a search complexity of  $O(d \log N)$ , where  $N$  is the number of training samples. With our training dataset of 417 messages, this logarithmic complexity ensures efficient classification operations. This efficiency is particularly relevant given that real-world LoRaWAN deployments typically have low message rates due to duty cycle restrictions and the nature of IoT applications.
- **Message processing rate:** In practical LoRaWAN deployments, the message processing requirements are manageable due to:
  1. LoRaWAN duty cycle restrictions (1% for uplinks in EU868 band);
  2. Typical IoT application patterns with periodic, rather than continuous, transmissions;
  3. The inherent characteristics of Class A LoRaWAN devices, which limit transmission frequency.
- **Model updates:** For dynamic conditions with mobile gateways, our implementation supports model retraining, as shown in Figure 6b, with the retraining process handling the modest dataset efficiently due to its logarithmic complexity.

These computational requirements proved sufficient for maintaining the 90.42% accuracy and 100% recall rates achieved in our edge computing experiments, even with gateway mobility, as shown in Figure 10. The system successfully processed all 595 messages in our test deployment, with the logarithmic complexity and naturally low message rates of LoRaWAN making it suitable for real-world applications with similar scales and mobility patterns.

#### 4.7. Limitations and Challenges

While our intrusion detection system demonstrated promising results in both centralized and edge computing environments, it is important to acknowledge the limitations of our study and the challenges encountered during experimentation. These constraints not only contextualize our findings but also highlight areas for future research and improvement.

A primary limitation of our study lies in the dataset characteristics. Although derived from real-world LoRaWAN deployments, our dataset was relatively small for each device, comprising 590 messages for the centralized environment and 595 for the edge environment. This limited sample size may impact the generalizability of our results to larger, more diverse LoRaWAN networks. Additionally, the intrusions in our dataset were simulated based on known attack patterns, which may not fully capture the subtleties of real-world attacks.

In our edge computing scenario, we simulated gateway mobility using discrete position changes rather than continuous movement. While this approach provided valuable insights into the challenges of mobile deployments, it may not fully represent the complexities of real-world mobile LoRaWAN applications. Future work should consider more diverse and complex movement patterns to better reflect real-world scenarios.



The performance metrics used in our study, while standard in machine learning evaluations, may have limitations in the context of network security. The high accuracy achieved, particularly in the centralized environment, could be partially attributed to class imbalance in our dataset. Future research should consider metrics that are more robust to class imbalance, such as the Matthews Correlation Coefficient (MCC). Furthermore, while we discussed potential latency benefits of edge computing, our current metrics do not directly measure or compare detection speed between the two environments.

Our edge computing implementation faced several challenges that warrant further investigation. Resource constraints of edge devices, including memory and processing power limitations, were not fully accounted for in our experiments. The energy consumption of running the IDS on possibly battery-powered edge gateways, a critical consideration for practical LoRaWAN deployments, was not evaluated. Additionally, our current implementation does not address the challenge of efficiently and securely updating the machine learning model across distributed edge devices.

The generalizability of our findings to different LoRaWAN configurations and environmental factors presents another limitation. Our experiments were conducted with specific LoRaWAN settings, and the effectiveness of the IDS across different spreading factors, device types, and classes requires further investigation. Moreover, the impact of various environmental factors, such as urban versus rural settings or indoor versus outdoor deployments, on the IDS performance was not extensively explored in this study.

Addressing these limitations and challenges presents opportunities for future research and improvements to our LoRaWAN intrusion detection system. Despite these constraints, the current results provide valuable insights into the potential of machine learning-based intrusion detection for both static and mobile LoRaWAN deployments. Future work should focus on expanding the dataset, developing more sophisticated mobility models, investigating the impact of different LoRaWAN configurations and environmental factors, and addressing the specific challenges of edge computing in IoT security contexts.

## 5. Conclusions

This study presents a novel and effective approach to intrusion detection in LoRaWAN networks using machine learning techniques, successfully implemented and evaluated in both centralized and edge computing environments. Our findings demonstrate the robust capability of the proposed Intrusion Detection System (IDS) in identifying potential security threats, achieving impressive accuracy and recall rates in both static and mobile scenarios. The centralized approach exhibited exceptional performance, with 99.32% accuracy and an F1 score of 95.24%, while the edge computing implementation demonstrated strong resilience under mobile conditions, achieving 90.42% accuracy and an F1 score of 92.59%. These results strongly validate the potential of machine learning-based intrusion detection for enhancing LoRaWAN security, particularly in diverse and dynamic IoT deployments.

Our research has identified valuable opportunities for further advancement of this promising technology. While the edge computing implementation shows strong potential, particularly in handling mobile scenarios, there are opportunities to enhance its precision beyond the current 86.21% to match the centralized environment's 90.91%. This improvement potential highlights the exciting possibilities for developing more sophisticated algorithms that are specifically tailored to mobile LoRaWAN deployments.

The successful implementation of our system provides a strong foundation for scaling to larger LoRaWAN deployments. Our edge computing approach has demonstrated the viability of distributed processing, opening up promising avenues for handling extensive IoT networks efficiently. The insights gained from our implementation with real-world data, though currently focused on specific device samples, provide valuable guidance for expanding to more diverse deployment scenarios.

Building on these successful outcomes, we identify several promising directions for future research:

- **Enhanced mobility support:** Development of advanced machine learning techniques to further optimize performance in mobile LoRaWAN deployments, building on our current strong results to achieve even higher precision in dynamic environments.
- **Scalability advancement:** Exploration of innovative hierarchical and distributed processing architectures to extend our successful approach to larger-scale LoRaWAN deployments while maintaining the high detection accuracy demonstrated in this study.
- **Resource optimization:** Refinement of our implementation for edge devices, focusing on maintaining high performance while optimizing resource utilization, building on our current effective edge computing approach.
- **Extended validation:** Expansion of our successful testing approach to include a broader range of deployment scenarios and device types, further validating and enhancing our system's effectiveness.
- **Environmental adaptation:** Investigation of performance optimization across various deployment environments, leveraging our current strong results to develop even more adaptive and robust solutions.
- **Continuous improvement:** Development of efficient methods for model updates in distributed environments, ensuring our system remains effective as threats evolve.

In conclusion, this work contributes to IoT security by demonstrating the effectiveness of machine learning-based intrusion detection for LoRaWAN networks in both centralized and edge computing paradigms. Our results show that high-performance intrusion detection is achievable in both static and mobile scenarios, providing a strong foundation for securing modern IoT networks. As IoT deployments continue to expand and evolve, the insights and methodologies developed in this study offer valuable guidance for the continued advancement of security measures for LoRaWAN and similar LPWAN technologies. The promising results and identified opportunities for enhancement point to an exciting future in IoT network security, where even more robust and adaptable solutions can be developed building on this work.

**Author Contributions:** Conceptualization, N.C., G.E. and F.F.; methodology, G.E.; software, F.F. and G.E.; validation, G.E.; investigation, G.E. and F.F.; resources, N.C., J.S. and F.F.; data curation, G.E.; writing—original draft preparation, G.E.; writing—review and editing, N.C.; visualization, G.E.; supervision, N.C. and J.S.; project administration, N.C.; funding acquisition, N.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by FEDER—Açores2020, within the project Toursignal, ref. ACORES-01-0247-FEDER-000028, and by Fundação para a Ciência e a Tecnologia through LASIGE, ref. UIDB/00408/2020 and ref. UIDP/00408/2020.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data is contained within the article.

**Acknowledgments:** The authors thank the Future Internet Technologies team at Instituto Superior de Engenharia de Lisboa, Instituto Politécnico de Lisboa, involved in this project, for their support and knowledge.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Mekki, K.; Bajic, E.; Chaxel, F.; Meyer, F. A comparative study of LPWAN technologies for large-scale IoT deployment. *ICT Express* **2019**, *5*, 1–7. [[CrossRef](#)]
2. Paul, B. An Overview of LoRaWAN. *Wseas Trans. Commun.* **2021**, *19*, 231–239. [[CrossRef](#)]
3. Bouziani, O.; Benaboud, H.; Chamkar, A.S.; Lazaar, S. A Comparative Study of Open Source IDSs According to Their Ability to Detect Attacks. In Proceedings of the 2nd International Conference on Networking, Information Systems & Security, Rabat, Morocco, 27–29 March 2019.
4. Khraisat, A.; Alazab, A. A critical review of intrusion detection systems in the internet of things: Techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. *Cybersecurity* **2021**, *4*, 18. [[CrossRef](#)]

5. Cuomo, F.; Garlisi, D.; Martino, A.; Martino, A. Predicting LoRaWAN Behavior: How Machine Learning Can Help. *Computers* **2020**, *9*, 60. [[CrossRef](#)]
6. Sicato, J.; Singh, S.K.; Rathore, S.; Park, J. A Comprehensive Analyses of Intrusion Detection System for IoT Environment. *J. Inf. Process. Syst.* **2020**, *16*, 975–990. [[CrossRef](#)]
7. Abdaljabar, Z.H.; Ucan, O.N.; Ali Alheeti, K.M. An Intrusion Detection System for IoT Using KNN and Decision-Tree Based Classification. In Proceedings of the 2021 International Conference of Modern Trends in Information and Communication Technology Industry (MTICTI), Sana'a, Yemen, 4–6 December 2021; pp. 1–5. [[CrossRef](#)]
8. Mudgerikar, A.; Sharma, P.; Bertino, E. Edge-Based Intrusion Detection for IoT Devices. *ACM Trans. Manage. Inf. Syst.* **2020**, *11*, 18. [[CrossRef](#)]
9. Albulayhi, K.; Smadi, A.A.; Sheldon, F.T.; Abercrombie, R.K. IoT Intrusion Detection Taxonomy, Reference Architecture, and Analyses. *Sensors* **2021**, *21*, 6432. [[CrossRef](#)]
10. Ullah, I.; Mahmoud, Q.H. A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks. In *Proceedings of the Advances in Artificial Intelligence*; Goutte, C., Zhu, X., Eds.; Springer: Cham, Switzerland, 2020; pp. 508–520.
11. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [[CrossRef](#)]
12. Mohy-eddine, M.; Guezzaz, A.; Benkirane, S.; Azrou, M. An efficient network intrusion detection model for IoT security using K-NN classifier and feature selection. *Multimed. Tools Appl.* **2023**, *82*, 23615–23633. [[CrossRef](#)]
13. Baz, M. SEHIDS: Self Evolving Host-Based Intrusion Detection System for IoT Networks. *Sensors* **2022**, *22*, 6505. [[CrossRef](#)]
14. Spadaccino, P.; Cuomo, F. Intrusion Detection Systems for IoT: Opportunities and challenges offered by Edge Computing and Machine Learning. *arXiv* **2022**. [[CrossRef](#)]
15. Zoppi, T.; Ceccarelli, A.; Puccetti, T.; Bondavalli, A. Which algorithm can detect unknown attacks? Comparison of supervised, unsupervised and meta-learning algorithms for intrusion detection. *Comput. Secur.* **2023**, *127*, 103107. [[CrossRef](#)]
16. Mohamed, A.; Wang, F.; Butun, I.; Qadir, J.; Lagerström, R.; Gastaldo, P.; Caviglia, D.D. Enhancing Cyber Security of LoRaWAN Gateways under Adversarial Attacks. *Sensors* **2022**, *22*, 3498. [[CrossRef](#)] [[PubMed](#)]
17. Danish, S.M.; Nasir, A.; Qureshi, H.K.; Ashfaq, A.B.; Mumtaz, S.; Rodriguez, J. Network Intrusion Detection System for Jamming Attack in LoRaWAN Join Procedure. In Proceedings of the 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, USA, 20–24 May 2018; pp. 1–6. [[CrossRef](#)]
18. Liao, Y.; Vemuri, V. Use of K-Nearest Neighbor classifier for intrusion detection11An earlier version of this paper is to appear in the Proceedings of the 11th USENIX Security Symposium, San Francisco, CA, August 2002. *Comput. Secur.* **2002**, *21*, 439–448. [[CrossRef](#)]
19. Mari, A.G.; Zinca, D.; Dobrota, V. Development of a Machine-Learning Intrusion Detection System and Testing of Its Performance Using a Generative Adversarial Network. *Sensors* **2023**, *23*, 1315. [[CrossRef](#)]
20. Ferrag, M.A.; Maglaras, L.; Ahmim, A.; Derdour, M.; Janicke, H. RDTIDS: Rules and Decision Tree-Based Intrusion Detection System for Internet-of-Things Networks. *Future Internet* **2020**, *12*, 44. [[CrossRef](#)]
21. Shah, R.A.; Qian, Y.; Kumar, D.; Ali, M.; Alvi, M.B. Network Intrusion Detection through Discriminative Feature Selection by Using Sparse Logistic Regression. *Future Internet* **2017**, *9*, 81. [[CrossRef](#)]
22. Syamsuddin, I.; Barukab, O.M. SUKRY: Suricata IDS with Enhanced kNN Algorithm on Raspberry Pi for Classifying IoT Botnet Attacks. *Electronics* **2022**, *11*, 737. [[CrossRef](#)]
23. Gyamfi, E.; Jurcut, A. Intrusion Detection in Internet of Things Systems: A Review on Design Approaches Leveraging Multi-Access Edge Computing, Machine Learning, and Datasets. *Sensors* **2022**, *22*, 3744. [[CrossRef](#)]
24. Zao, J.; Byers, C.; Murphy, B.; AbiEzzi, S.; Banks, D.; An, K.; Michaud, F.; Bartfai-Walcott, K. *The Industrial Internet of Things Distributed Computing in the Edge*; Technical Report; Industrial Internet Consortium: Boston, MA, USA, 2020.
25. Alsubhi, K. A Secured Intrusion Detection System for Mobile Edge Computing. *Appl. Sci.* **2024**, *14*, 1432. [[CrossRef](#)]
26. Stančin, I.; Jović, A. An overview and comparison of free Python libraries for data mining and big data analysis. In Proceedings of the 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 20–24 May 2019; pp. 977–982. [[CrossRef](#)]
27. Baak, M.; Koopman, R.; Snoek, H.; Klous, S. A new correlation coefficient between categorical, ordinal and interval variables with Pearson characteristics. *Comput. Stat. Data Anal.* **2020**, *152*, 107043. [[CrossRef](#)]
28. Waleed, A.; Jamali, A.F.; Masood, A. Which open-source IDS? Snort, Suricata or Zeek. *Comput. Netw.* **2022**, *213*, 109116. [[CrossRef](#)]
29. Albin, E.; Rowe, N.C. A Realistic Experimental Comparison of the Suricata and Snort Intrusion-Detection Systems. In Proceedings of the 2012 26th International Conference on Advanced Information Networking and Applications Workshops, Fukuoka, Japan, 26–29 March 2012; pp. 122–127. [[CrossRef](#)]
30. Pritz, S.; Zeinzinger, M.; Praschl, C.; Krauss, O.; Harrer, M. Performance Impact of Parallel Access of Time Series in the Context of Relational, NoSQL and NewSQL Database Management Systems. In Proceedings of the 2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Tenerife, Canary Islands, Spain, 19–21 July 2023; pp. 1–6. [[CrossRef](#)]
31. Kramer, O. Scikit-Learn. In *Machine Learning for Evolution Strategies*; Springer International Publishing: Cham, Switzerland, 2016; pp. 45–53. [[CrossRef](#)]

- 
32. Imam, F.; Musilek, P.; Reformat, M.Z. Parametric and Nonparametric Machine Learning Techniques for Increasing Power System Reliability: A Review. *Information* **2024**, *15*, 37. [[CrossRef](#)]
  33. Cunningham, P.; Delany, S.J. k-Nearest Neighbour Classifiers—A Tutorial. *ACM Comput. Surv.* **2021**, *54*, 128. [[CrossRef](#)]

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.