Chapter 4

Machine Learning-Based Interference Mitigation in Long-Range Networks

The previous chapter presented a federated learning approach to simultaneously enhance communication and provide robustness against the imperfect labels for the long range based transportation system. This chapter considers challenges of wireless communication in a high-ceiling building. It aims to enhance the performance of the LoRa network in smart buildings and reduce maintenance costs, time, and interruptions caused by faulty sensors.

4.1 Introduction

Smart building utilizes sensors, acoustics, and networking devices to collect and analyze data about complex buildings, and it automatically makes decisions based on this data. Depending on the size of the building, a smart building can be classified as small, medium, or large [75]. Small and medium-sized buildings have a limited number of devices and smaller distances between them for environmental monitoring, making maintenance simpler than in large-sized smart buildings with a vast number of devices and larger distances among them. Monitoring buildings with high ceilings, such as airport terminal structures, large workshops, warehouses, factories, malls, auditoriums, indoor stadiums, railway stations, and theatres, is more challenging as they require frequent maintenance of special safety equipments [76, 77].

Smart buildings are becoming more prevalent as they offer several benefits such as improved energy efficiency, cost savings, and increased comfort for occupants. One

critical aspect of a smart building is a monitoring system that uses various types of sensors to monitor the health of building structures [78]. However, the placement of such sensors in high-ceiling smart buildings is cost-effective and challenging. The accuracy of the data collection mainly depends on the placement of the sensors. In addition, the frequent replacement of batteries or maintenance of such sensors becomes infeasible as it interrupts the running application of the building area. For example, work on the roof of a public building, such as an airport or railway station, requires the evacuation of the area. Thus, a sensor is considered suitable for the smart building if it requires less maintenance, battery replacement, and failure. To ensure the efficient and effective functioning of a smart building monitoring system, it is essential to carefully select the appropriate sensors and strategically place them [75].

The exchange of sensory data among smart building devices can be accomplished through various wireless communication techniques. Short-range techniques such as WiFi, BLE, and Zigbee, among others, consume less energy when communicating sensory data to nearby devices [79–81]. However, long-range communication techniques such as 5G and LTE consume huge amounts of energy when transferring data over long distances. Despite their different advantages, both short-range and long-range communication techniques are unsuitable for smart buildings [82]. The former is because of limited communication range, while the latter is because of high power consumption. Additionally, the short-range techniques cannot transfer sensory data from sensors attached at the top of high-ceiling smart buildings, which poses a significant challenge. Hence, there is a need for more suitable wireless communication techniques that are energy-efficient and can transmit data over long distances.

The Long Range Wide Area Network (LoRaWAN) utilizes a star-of-stars network topology and medium access control to provide low power and long-range communication, making it an ideal wireless communication for high-ceiling smart buildings [83]. The benefits of LoRaWAN include its long communication range, low power consumption, and ease of implementation. These advantages have resulted in exponential growth in the use of LoRaWAN applications in smart buildings, particularly in high-ceiling smart buildings where the distance between the sensors and devices is higher. The LoRaWAN architecture consists of several entities, including LoRa Nodes (LNs), LoRa Gateway (LG), Network Server (NS), and Application Server (AS). LNs are embedded with sensors to monitor the health of the buildings, and they transfer the sensory data to the LG using LNs. The LG transfers the data to the AS via NS [24, 27].

LoRa technology has several advantages in smart buildings, but it also presents challenges too. LoRa employs virtual channels known as Spreading Factors (SFs) to

transfer sensory data between LNs and LG. LoRa supports a limited number of virtual channels, which means that an LG can only connect to a limited number of LNs on a given SF. Interference is a common issue in LoRa, especially when multiple LNs simultaneously attempt to transmit data to a single LG on a given SF. This interference problem is even more severe when sensing and communicating health information of building structures, as it can prevent an LN from transmitting its critical data LG [84].

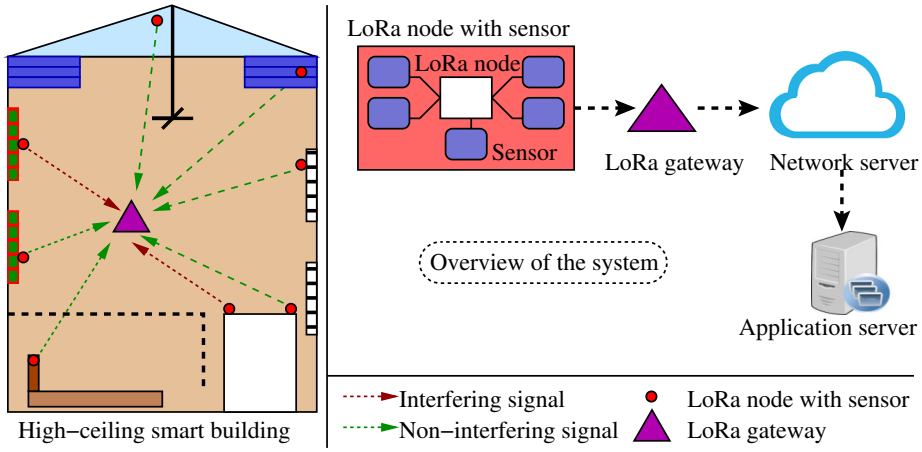


Figure 4.1: Overview of system model with the illustration of interference problem in LoRa network.

In this chapter, we present a novel solution to the challenges of wireless communication in a high-ceiling building, as illustrated in Figure 4.1. Conventional short-range communication technologies are not suitable due to the large distance between the sensors attached to the ceiling and the network devices on the ground, while long-range communication techniques are not viable due to their high energy consumption. To overcome these challenges, we propose the use of LoRa, a low-power and long-range wireless communication technology. In our scenario, LNs are attached to the ceiling to sense the health of building structures, and an LG is used to collect and transmit the sensory data to the NS for further processing. This scenario poses challenges, such as:

1. **Interference in the network:** LoRa's limited virtual channels can cause interference when multiple LNs attempt to communicate with a LG on a given SF.

2. **Identifying faulty sensors:** Ensuring the proper functioning of sensors is critical for the health and safety of building occupants but identifying and maintaining faulty sensors can be costly and time-consuming. Quick identification of faulty sensors from a large number of sensors is challenging, and delays in detection result in financial loss or even endanger human life. Thus, efficient sensor fault

detection and maintenance is a crucial challenge in the context of smart buildings.

3. **Energy efficiency:** Ensuring energy efficiency is the main challenge in the network of smart buildings since the sensors, which are powered by batteries, cannot be frequently replaced or recharged.

4. **Cost-effectiveness:** To ensure a cost-effective solution for a large-size smart building, it is crucial to have a large number of sensors that are low-cost.

To overcome the aforementioned challenges in the scenario, this chapter presents a novel machine learning-based solution to identify and mitigate interference issues in the LoRa network of a high-ceiling smart building. Specifically, we aim to answer the following question: ***how can we quickly identify the number of LNs causing interference in the LoRa network of a smart building?*** Our proposed approach involves analyzing network parameters of received signals from LNs to LG. It includes data collection, preprocessing, and training of a learning model to build a classifier that predicts the number of interfering LNs and identifies them accurately. By leveraging our solution, we aim to enhance the performance of the LoRa network in smart buildings and reduce maintenance costs, time, and interruptions caused by faulty sensors.

4.1.1 Major contributions

To the best of our knowledge, this is the first study to develop a machine-learning-based approach for identifying the LNs that cause interference in high-ceiling smart buildings. Our approach does not require any additional hardware and works on the LG, making it a cost-effective solution. Following are the major contributions of this work:

- We introduce a novel approach that utilizes a classification model to identify interfering LNs in high-ceiling smart buildings. Our approach involves collecting and analyzing network parameters such as signal-to-noise ratio (SNR) and received signal strength indicator (RSSI) from the signals to extract features for the classifier. The approach classifies interference based on the number of interfering LNs, where each class represents a different number of interfering LNs. Additionally, we propose a push-based mechanism for detecting and adjusting the power levels of faulty LNs to mitigate interference. Our approach is hardware-independent, making it a cost-effective solution that can be implemented on the LG platform. We contribute to the field in two ways: (1) *a novel classification approach for identifying interfering LNs*, and (2) *a push-based mechanism for identifying faulty LNs*

and adjusting their power levels.

- Our next contribution is the creation of an open dataset of LoRa interference. This labeled dataset includes various deployment scenarios with multiple LNs, both with and without interference. For each message sent by the LN to the LG, the dataset includes important features such as SNR, RSSI, payload, and timestamp. The free availability of this dataset will enable researchers to compare the performance of different interference mitigation techniques and encourage the development of new approaches for LoRa interference.
- Finally, we present a testbed that validates the feasibility of our approach. The testbed consists of low-cost in-house developed devices and modifies the existing LoRa device-to-device LMIC library to implement the approach. Our system achieves an average accuracy of approximately 98.85%, demonstrating its high performance. Our results highlight the system's ability to effectively detect faulty and interference devices, providing an advantage over the state-of-the-art.

4.1.2 Background and motivation

4.1.2.1 Background

The interference problem in LoRaWAN has been addressed using various approaches, including the use of RSSI-based SF allocation schemes. However, in real-world deployments, RSSI values can vary significantly due to obstacles, distance, and the presence of other LoRa signals, among other factors. This is particularly challenging in high-ceiling smart buildings where signals can experience significant attenuation and multipath fading. As a result, RSSI-based SF allocation needs to be more competent to solve the problem of interference in LoRaWAN. Recently, authors in [85] proposed a solution to the interference problem by using the CSMA protocol. Although CSMA has shown promise in addressing interference, it is also facing the issue of high delay. Moreover, it does not solve the problem of collisions when a random node joins and tries to send data simultaneously, which can further worsen the interference problem. Therefore, there is still a need for more effective and efficient solutions to the interference problem in LoRaWAN, especially in challenging environments like high-ceiling smart buildings.

Several papers have analyzed the performance of the LoRa network, with a focus on improving network performance. In particular, many of these studies have proposed different SF allocation and channel orthogonalization techniques to optimize the network. For instance, the paper uses a distance-based SF allocation approach and channel orthogonalization with different SF, CR, and bandwidth. In recent work, authors in [86]

proposed solving the interference problem in LoRaWAN using game theory techniques. Specifically, the authors used Bayesian and Stackelberg games to address the interference issue. Other studies, such as [70,87–91], have examined the scalability of the LoRa network, particularly in relation to distance-based SF allocation techniques. The authors noted that as the network scales up, the interference problem. They highlighted the importance of SF scheduling in improving the scalability of the network. These studies demonstrate the ongoing efforts to optimize the performance of LoRa networks and address their challenges, such as interference and scalability.

Optimizing SF allocation in LoRa networks is a challenging task, as it involves addressing issues such as the capture effect and fairness considerations. Several approaches have been proposed in the literature, with a common thread being the use of *game theory* techniques to model the behavior of LoRa users and channels. In particular, some recent work has proposed a game-theoretic framework for SF allocation, where LoRa users and channels are characterized as greedy players that aim to maximize their utility. By formulating SF allocation as a many-to-one matching game, this framework can account for capture effect and fairness concerns. To improve the performance of LoRa networks, the authors in [92–94] recently proposed a sequential strategy for allocating SF to LN. This approach balances SF allocation across the LoRaWAN and takes advantage of RSSI information to improve resource allocation. These studies demonstrate the use of game theory and other techniques to address the challenges of SF allocation in LoRa networks. By improving SF balancing and resource allocation, it is possible to enhance the performance and fairness of LoRaWAN.

The authors in [95] introduced a Multi-Stream Orthogonal Network Decoupling protocol, MS-OND, for wireless communication systems with multi-antenna source and destination nodes. MS-OND also comprised some single-antenna half-duplex relay nodes. The primary motivation of the work was to successfully deliver multiple data streams for each multi-antenna source-destination pair by exploiting multiuser diversity gain in fading channels. The authors identified that the proposed MS-OND protocol is suitable for scenarios like massive Machine-Type Communications (mMTC) and the Internet of Things (IoT) in 5G wireless networks. It can be applied to low-cost devices with a half-duplex and single-antenna configuration, making them potential candidate relay nodes. Next, the authors in [96] highlighted the role of multi-hop Device-to-Device (D2D) communications in underlying cellular networks.

Apart from the objective of [95], here the authors tried to effectively address the increasing demand for Internet access from mobile users. They concluded that the direct single-hop D2D communication mechanism has limitations in delivering quality

transmissions over a large area. In addition, they investigated the impact of interference and network traffic conditions on the quality of D2D communications and derives analytical expressions for end-to-end packet loss probability (E2EPLP) in the presence or absence of XOR coding.

Further, Behnad *et al.* [97] has thoroughly analyzed the performance of opportunistic relaying in a dual-hop Amplify-and-Forward (AF) relay network. The authors have considered the scenario, where the relaying nodes are distributed according to a homogeneous two-dimensional Poisson point process with a fixed density. Such assumption of a random spatial distribution of relaying nodes not only distinguish the proposed work from other but also leads to more realistic results. The considered scenario is more realistic as the number of relays and their distances from the source and destination are typically unknown in practical cases. Finally, Zanella *et al.* [98] explored a scenario where multiple sources are transmitting messages to their intended destinations using relays in a decode-and-forward two-hop mechanism. The objective of the authors was to minimize interference using two opportunistic relay selection mechanisms. These mechanisms aimed to select relays that can minimize interference and improve overall performance. The authors analytically evaluated the performance of these selection mechanisms in terms of outage probability and average achievable rate. The analysis assumed that the relay nodes are distributed according to a Poisson point process, which is a commonly used mathematical model for random spatial distributions.

Furthermore, the authors in [99] discussed a substantial review of emerging trends in future smart grid research integrated with the technical work. The primary focus was on visualising an innovative smart grid that utilizes the power of artificial intelligence, IoT, and 5G networks. In addition, the authors have addressed the challenges inherent in building next-generation smart grids, particularly integrating AI, IoT, and 5G to boost smart grid technology. They also provided potential solutions to these challenges and suggested standards that can support this innovative direction. Similarly, the authors in [100] highlighted the role of adversarial attack methods in evaluating the robustness of deep learning-based classifiers, particularly in wireless signal classification. They introduced a real-world threat model that bridges the gap between idealized assumptions and real-world conditions in adversarial attack scenarios. By presenting an innovative IC-UAP crafting method and a physical attack algorithm, this approach enhances the effectiveness and applicability of malicious attacks against deep learning-based wireless signal classifiers.

4.1.2.2 Motivation

Existing work on interference in LoRa networks has largely focused on two types of SF allocation schemes: *distance-based* and *RSSI-based* [101]. However, in practice, only distance-based allocation may be needed to address interference and scalability concerns. One key limitation of the existing work is the lack of consideration for SNR pattern analysis in LoRa networks. SNR patterns provide a useful indicator of interference signals in the network, which can be particularly relevant in scenarios where multiple LNs of the same SF are located in close proximity and attempt to transmit data simultaneously. In such cases, only a subset of nodes may be able to send data to the LG, leading to data loss and the creation of interference signals.

This chapter focuses on addressing interference in LoRa networks using a novel approach based on real-time SNR pattern analysis. Specifically, we propose leveraging machine learning techniques to predict interference signals in the LoRa network, providing a more effective means of addressing this challenge. By analyzing SNR patterns in real-time, our approach can help to identify and address interference issues quickly and efficiently, leading to improved network performance and reliability.

The rest of the chapteris structured as follows: Section 4.2 formally defines the notations, and assumptions, and presents an overview of the system. In Section 4.3, the proposed system is presented in detail. Section 4.4 presents the dataset collection and testbed results, and Section 4.5 concludes the chapterwith a discussion of the findings and future work.

4.2 Preliminaries and problem statement

This section presents the system model for the considered LoRaWAN network, along with the key terminologies and notations used throughout this work. Additionally, we draw attention to the issue of co-SF interference in the network.

4.2.1 Preliminaries

4.2.1.1 LoRa and LoRaWAN

Long-Range Wide Area Network (LoRaWAN) is a widely adopted protocol in the Low-Power WANs (LPWANs) family. It facilitates a long communication range while consuming low power. LoRa Alliance is responsible for providing public specifications and promotion for LoRaWAN. The LoRaWAN uses IEEE 802.15.4 at the physical or radio layer, also known as Long-Range (LoRa). We can use conventional network protocols

above the LoRa physical layer [1]. However, the LoRa alliance recommends the LoRaWAN protocol with radios having the duty cycle $\leq 1\%$. The working principle of the LoRa modulation technique relies upon the Chirp Spread Spectrum (CSS). LoRa encodes data via frequency chirps with linear frequency variations over time. Chirp modulation is a technique of transmitting symbols, where symbols are encoded into multiple signals of decreasing (down-chirp) or increasing (up-chirp) radio frequencies.

4.2.1.2 LoRa communication

LoRa supports a flexible communication range using different Spreading Factors (SFs), *i.e.,* SF 7, SF 8, SF 9, SF 10, SF 11, and SF 12. The lower SF (*e.g.,* SF 7) possesses a high data rate, less communication range, and low packet loss and vice-versa for higher SF (*e.g.,* SF 12). LoRaWAN defines an Adaptive Data Rate (ADR) scheme by the network server to control the uplink transmission of LN. A network server optimizes data rate and transmission power using the ADR bit. LoRaWAN uses three signal bandwidths, *i.e.,* 125 kHz, 250 kHz, and 500 kHz. Furthermore, LoRaWAN uses the logic of the Media Access Control (MAC) protocol specification of the data link layer, which allows the end node to exchange information with the network server through a gateway (or relay). After getting a message from the end node, the network server sends that message to the application server, which is directly connected to the users.

4.2.1.3 LoRa encryption and parameters

LoRaWAN uses two Advanced Encryption Standard (AES), *i.e.,* AES-128, techniques to secure the data: 1) AES-128 security protocol between LN and NS and 2) AES-128 security protocol between the LN and AS. The parameters impacting communication via LoRa are the spreading factor, bandwidth, coding rate, and transmission power. These parameters impacted the received signal strength, power consumption, and coverage range. Table **??** illustrates the LoRa communication parameters with their corresponding symbols and range of values.

Further, the three factors that affect data rate, d_r, in the networks are b, s, and c^r. The expression for d_r is expressed as follows:

$$d_r = b \times \frac{s}{2^s} \times c^r. \tag{4.1}$$

The symbol duration using spreading factor and bandwidth is expressed as follows:

$$T_{sym} = \frac{2^s}{b}. \tag{4.2}$$

4.2.1.4 Interference node prediction using SVM

There is a separate SNR limit for each SF. LoRa network's receiver sensitivity determines the communication range.

$$\text{Rx sensitivity} = -174 + 10log_{10}(b) + NF + SNR(limit), \tag{4.3}$$

where NF denote the noise factor. $SNR(limit)$ value decreases, receiver sensitivity decreases, and communication range increases, as shown in Table 4.1. SNR value fluctuates when the number of LN increases or decreases. Fluctuation is more visible when the number of simultaneous messages sends by the LN to the LG. We collect different combination data in SQM dataset. We apply SVM techniques to predict the number of interference nodes in the LoRa network. Using SVM techniques we analyse the pattern of SNR fluctuation to predict the interference LNs.

The SNR can be further defined as $\frac{P_{tx}A_0gr^{-\alpha}}{\sigma^2}$, where P_{tx} denotes the transmit power of transmission, σ^2 denotes the noise power, g denotes their corresponding channel fading power and r denotes the distance between the gateway and LoRa node. LoRa network, LoRa messages decode at the gateway based on signal strength. Additionally, A_0 is the Friis transmission equation defined as $A_0 = (\frac{c}{4\Pi f_c})^2$, where the Friis transmission equation comes with carrier frequency f_c and velocity of light c.

Table 4.1: Illustration of different SFs with their SNR limit, receiver sensitivity, and range.

SF	**SNR (LIMIT)**	**Receiver sensitivity**	Range
7	-6	-123	$a_0 - a_1$
8	-9	-126	$a_1 - a_2$
9	-12	-129	$a_2 - a_3$
10	-15	-132	$a_3 - a_4$
11	-17.5	-134.5	$a_4 - a_5$
12	-20	-137	$a_5 - a_6$

4.2.1.5 Receiver sensitivity with shadowing/fading Factor

The receiver sensitivity defined in (4.3) is simpler and utilized during the result analysis. In this section, we describe the more complicated expression for receiver sensitivity considering the shadowing/fading factor. To incorporate shadowing or fading factors into the equation ((4.3)) for Rx sensitivity, we have to employ stochastic terms that represent the random variations caused by these effects. Here, Shadowing refers to large-scale signal variations due to obstacles or other physical factors while fading refers to

small-scale signal variations caused by multi-path propagation.

In this work, we have added random variables to the equation ((4.3)) that follow specific distributions representing shadowing and fading. The expression is as follows:

$$\text{Rx sensitivity} = -174 + 10\log_{10}(b) + NF + SNR(\text{limit}) + X_{\text{shadow}} + X_{\text{fade}}.$$

where X_{shadow} is the shadowing term (shadow fading), a random variable following a log-normal distribution and X_{fade} is the fading term (small-scale fading), a random variable following a Rayleigh distribution.

However, the distribution may be varied as the specific distribution of the shadowing and fading terms depends on the characteristics of the environment and the wireless channel. In this work, we employ log-normal distribution for shadowing, while the Rayleigh or Rician distributions for fading in different scenarios. Further, adding shadowing and fading terms makes the equation probabilistic; thus, we have to analyze the statistical behaviour of the system rather than looking for deterministic solutions. This kind of analysis resembles that of wireless communication systems to assess their performance under real-world conditions.

4.2.2 Overview of network model

This work considers a LoRaWAN network scenario comprising N LNs and a single LG. The network uses the uplink model for this single LG network and is assumed to possess co-SF interference. The interference of different SFs is beyond the scope of this work. We consider multiple sensors connected to an LN to sense environmental activities like temperature, humidity, wind speed, air quality, *etc.* Each LN transfers the collected sensory data to the LG using the LoRaWAN protocol. Later, LG transmits the received data to NS using conventional internet protocols. NS processes the received data for making circumstance-specific decisions such as early warning alarms. Afterwards, NS conveyed these processed results to the AS via conventional internet protocols. Finally, the users can easily access the results from the AS.

4.2.3 Interference computation

When multiple nodes transmit data simultaneously to the gateway, it generates network interference. This thesis presents an innovative method to identify interfering Long-Range Nodes (LNs) using a machine learning-based classification model. Our approach involves gathering and analyzing network parameters, such as signal-to-noise ratio and received signal strength indicator, from the signals to extract features that the classifier

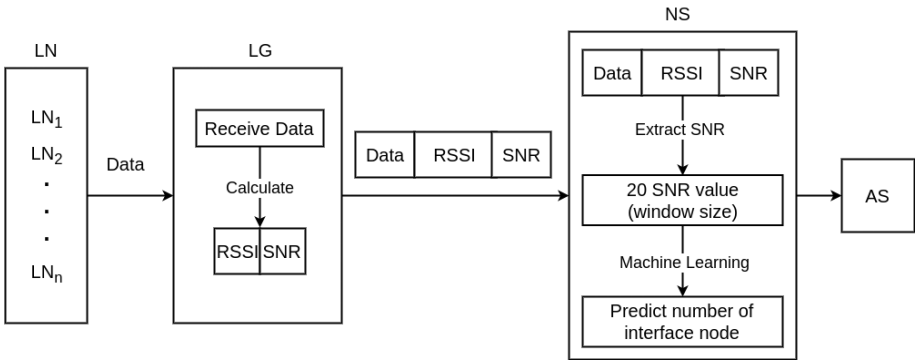


Figure 4.2: Flow diagram of our proposed model.

uses for interference classification. The approach categorizes interference based on the number of interfering LNs, with each class representing a distinct number of interfering LNs.

4.2.4 Problem statement

LoRa uses the random access control protocol, *i.e.*, unslotted ALOHA for access control; thus, prone to signal interference. This simultaneous transmission of data from multiple LNs introduced the problem of signal interference; therefore, we only receive data from one LN on LG, in Figure 4.2. The interference problem becomes more complicated in the dense LoRa network, where an LG connects multiple LNs. The inference leads to message loss in the network. If we can detect the number of interfering signals or LNs, then we can provide an adequate mechanism to overcome message loss.

4.3 LoRa devices interference system

In this section, we first discuss the device designs used during the experiment. We next discuss the deployment of the devices for creating the dataset and testing the accuracy of the approach. Next, this section presents the LoRa Devices Interference (LDI) system to recognize the LDs which create interference in the network of the smart buildings. Figure 4.2 illustrates the block diagram of the LDI system. In the data collection step, an LG collects the SNR and RSSI with a time-stamp of messages from LNs. The collected data then work as input for the preprocessing step where the raw data are windowing. The windowed and preprocessed data finally process to reorganize the LDs, creating an interference issue.

4.3.1 Design of LNs and LG

The approach uses LNs and LG to identify the interference issue, as shown in Figure 4.2. We design the LNs using the available micro-controller, LoRa shield, and sensors. The motivations for developing LNs are to reduce the cost, making them based on the requirement, and suitable for India located free frequency spectrum. The components of the LN are Arduino nano, LoRa shield, sensors, and a $5V$ battery. We select Arduino nano because it has 14 input/output digital pins, a crystal oscillator of 16 MHz, an operating voltage that varies from $5V$ to $12V$, supports the serial protocol, and has a mini USB Pin upload code and charges the battery. The LoRa shield is attached to a micro-controller. The $SX1276$ Shield is attached with Arduino to send data and reach extremely long ranges at low data rates. It provides ultra-long-range spread spectrum communication and high interference immunity while minimizing current consumption. Finally, we use accelerometers and vibration sensors, attached to LNs.

The main objective of the chapteris to handle the interference. We design a single channel LG for facing interference whenever multiple LNs simultaneously transfer the data. Another motivation for developing single-channel LG is to make it a low-cost and $5V$ battery-powered device. Such LG can easily be deployed in the field. The LG consists of Raspberry Pi, LoRa shield, display, and laptop connection.

4.3.2 Dataset creation for the training of the system

We created the dataset by considering SNR and RSSI as parameters, different locations, number of LNs, and size of the payloads. The dataset is available at IEEE dataport [102].

4.3.2.1 Deployment scenario

To illustrate the impact of the obstacles, we consider indoor and outdoor scenarios. We consider the department of CSE IIT(BHU) building as an indoor building and the railway platform as an outdoor scenario. The floor map and the locations of the devices are shown in Figure 4.3. Here, we use single-channel LG, and devices change from 8 to 15. The distance between LNs and LG varies from 5 to 50 meters. The floor map illustrates the walls, doors, and windows between LNs and LG. We consider railway stations for the outdoor environment. The outdoor environment did not consist of obstacles between LNs and LG. The number of LNs varies from 5 to 100 meters, and the number of LNs 8 to 15. The environment is considered in the evaluation of the proposed approach through the following aspects:

- *Sustainable Design:* The system prioritizes low-power communication technologies like LoRa, which significantly reduce energy consumption compared to traditional methods, aligning with environmental sustainability goals.
- *Minimizing E-Waste:* The approach leverages existing hardware for implementation, thereby avoiding unnecessary electronic waste from deploying new hardware.
- *Adaptability to Environmental Factors:* The solution explicitly considers challenges posed by environmental factors such as obstacles (e.g., walls) and multipath fading in high-ceiling smart buildings. Experimental scenarios include indoor and outdoor deployments to ensure robustness under varied conditions.
- *Reducing Maintenance Impact:* By enabling accurate fault detection and interference mitigation, the system minimizes frequent sensor replacements and associated environmental disruptions, such as area evacuations in public spaces.
- *Efficient Use of Resources:* The adaptive mechanisms for power and spreading factor allocation optimize resource utilization, reducing environmental strain from excessive energy usage.

4.3.2.2 Network parameters

The designed devices support SF 7 to 12. We use 7 to 12 SFs with 0.3 kbps to 27 kbps data rate and $\frac{4}{5}$ to $\frac{4}{8}$ coding rate. We used 12 dbi transmission power during the experiment. While collecting the dataset, we consider the interference between 2 to 6 devices on a fixed SF. We called it different levels of interference. For creating the interference, two or more LNs simultaneously transmits the data to the LG.

The signal quality measure in the LoRa network is either the received signal strength indicator (RSSI) or SNR. RSSI changes over obstacles, distance, noise, and interference. Therefore, detecting interfering LNs in the LoRa Network via only RSSI is not feasible. On the other hand, SNR fluctuates primarily because of interference with other signals. LN away from LG may experience a high SNR value in a real-time environment, while LN near LG may experience a low SNR. SNR varies differently at the receiver's end (LG) for each SF, where the SNR limit for different SFs is non-identical. Thus, we can exploit SNR and RSSI values to predict the number of interfering LNs in the LoRa network. The dataset consisted time stamp when the message was received at LG, the SNR of each message with a different label of interference, and the RSSI values of the received message. Figure 4.2 summarized the selected dataset. Here, we illustrate the SNR and RSSI with interference for the different numbers of LNs, placement, and SFs. The dataset consists of two payloads, text messages and sensory data. The size of the payload is 10 and 60 bytes.

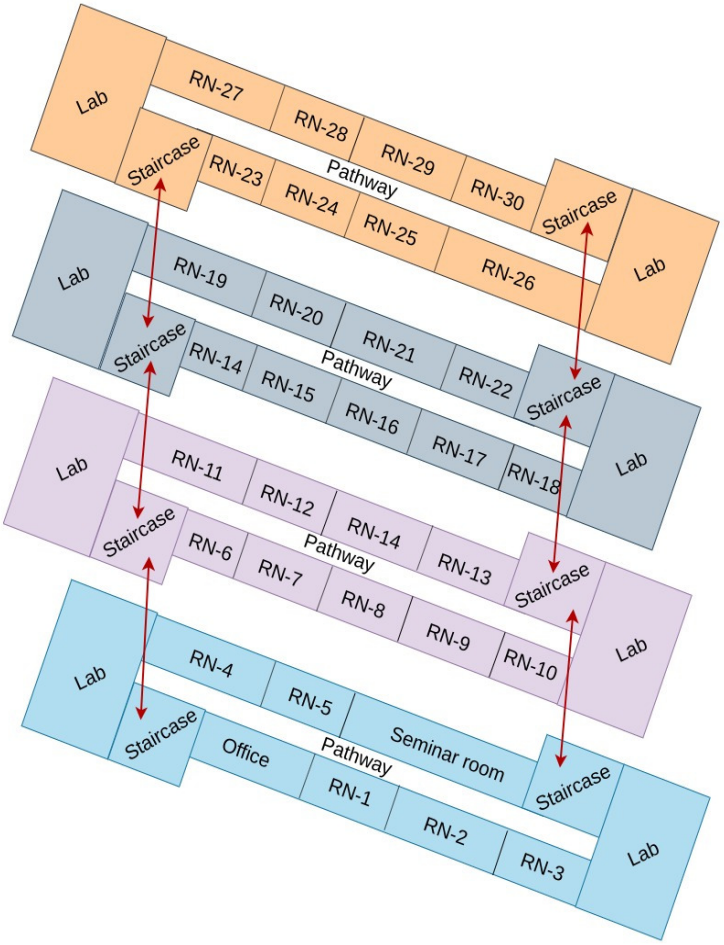


Figure 4.3: An overview of sensory deployment in our department for dataset collection.

4.3.3 Data collection for testing of the system

LoRa gateway receives data from LoRa nodes that transmit at different spreading factors (SF), which determine the transmission range and data rate. The gateway calculates the RSSI and SNR values of the received data, which are measures of the received signal strength and signal-to-noise ratio, respectively. These values are crucial in determining the quality of the received signal and the ability to decode the transmitted data. The LoRa gateway then sends the received data, along with the RSSI, SNR, and SF values, to the LoRa network server. The server stores the received data in different groups according to the spreading factors used. For example, data received at SF7 is stored in one group, while data received at SF8 is stored in another. This categorization is important because data transmitted at different spreading factors have different trade-offs between transmission range and data rate. Lower SFs (e.g., SF7)

allow for high data rates but have a shorter range, while higher SFs (e.g., SF12) allow for larger ranges but lower data rates. This organization of data can help in managing and analyzing the network's performance, as it facilitates the comparison of data transmitted at different SFs. Moreover, it can assist in troubleshooting and identifying issues that may arise in the network, as data received at different SFs may exhibit different characteristics or potential sources of interference.

4.3.4 Data filtering using joint Kalman filter

The joint Kalman filter is a variant of the Kalman filter that is used to estimate multiple state variables simultaneously. In our system, we used a Joint Kalman filter to estimate both the RSSI and SNR of a received signal.

Let x_k denote the state vector at time instant k. In our proposed work, the state vector will consist of two elements: 1) RSSI and 2) SNR. Thus, the measurement vector z_k will consist of the RSSI and SNR measurements at time instant k. The Joint Kalman filter proceeds in two stages, *i.e.,* prediction and update.

In the **prediction stage**, we use the state transition matrix F_k to predict the state vector at the next time instant, based on the state vector and process noise covariance matrix Q_k at the current time instant. Mathematically, it is given as:

$$x_{k|k-1} = F_k x_{k-1|k-1}, \tag{4.4}$$

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k. \tag{4.5}$$

In the **update stage**, we use the measurement matrix H_k and measurement noise covariance matrix R_k to update our estimate of the state vector based on the current measurement. The Kalman gain K_k is also computed in this stage.

$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}, \tag{4.6}$$

$$x_{k|k} = x_{k|k-1} + K_k (z_k - H_k x_{k|k-1}), \tag{4.7}$$

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}. \tag{4.8}$$

In this work, we use a $2x2$ identity matrix for both F_k and H_k, as we have two state variables and two measurements. The process noise covariance matrix Q_k and measurement noise covariance matrix R_k can be determined experimentally based on the characteristics of the system.

Example 1 *(**Numerical Example**) Let's illustrate a numerical example of the Joint Kalman filter for estimating RSSI and SNR of a received signal.*

Assume we have a measurement vector z_k of RSSI and SNR at time instant k as:

$$z_k = \begin{bmatrix} -75 & 20 \end{bmatrix}.$$

We start with an initial estimate of state vector $x_{0|0}$ and covariance matrix $P_{0|0}$ as:

$$x_{0|0} = \begin{bmatrix} -70 & 25 \end{bmatrix}, P_{0|0} = \begin{bmatrix} 10 & 10 \end{bmatrix}.$$

Assuming the process noise covariance matrix Q_k and measurement noise covariance matrix R_k are as follows:

$$Q_k = \begin{bmatrix} 0.01 & 0 & 0 & 0.01 \end{bmatrix},$$
$$R_k = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}.$$

We can define the state transition matrix F_k and measurement matrix H_k as 2×2 identity matrices:

$$F_k = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix},$$
$$H_k = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix}.$$

Assuming we have already calculated Kalman gain K_k as:

$$K_k = \begin{bmatrix} 0.498 & 0 & 0 & 0.498 \end{bmatrix}.$$

We can use the update stage equations to compute the new estimate of the state vector and covariance matrix:

$$x_{k|k} = x_{k|k-1} + K_k(z_k - H_k x_{k|k-1}),$$
$$x_{k|k} = \begin{bmatrix} -73.49 & 22.51 \end{bmatrix},$$
$$P_{k|k} = (I - K_k H_k) P_{k|k-1},$$
$$P_{k|k} = \begin{bmatrix} 5.01 & 0\ 0 & 5.01 \end{bmatrix}.$$

The new estimate of the state vector at time instant k is $x_{k|k}$, which gives us the estimated values of RSSI and SNR. We can continue this process at each time instant to update our estimate of the state vector and covariance matrix. Let us consider, we

have the following measurements of RSSI and SNR over time:

$$\begin{bmatrix} \textit{Time (k)} & \textit{RSSI Measurement} & \textit{SNR Measurement} \\ \textit{0} & \textit{-70} & \textit{20} \\ \textit{1} & \textit{-65} & \textit{22} \\ \textit{2} & \textit{-68} & \textit{21} \\ \textit{3} & \textit{-71} & \textit{18} \\ \textit{4} & \textit{-67} & \textit{19} \end{bmatrix}.$$

We can use the Joint Kalman filter to estimate the true values of RSSI and SNR, given these measurements. Let's assume that the initial state vector is:

$$x_0 = \begin{bmatrix} -75 & 15 \end{bmatrix}.$$

and the initial error covariance matrix is:

$$P_0 = \begin{bmatrix} 25 & 0 & 0 & 25 \end{bmatrix}.$$

We also assume that process noise covariance matrix is:

$$Q_k = \begin{bmatrix} 0.01 & 0\ 0 & 0.01 \end{bmatrix}.$$

The measurement noise covariance matrix is:

$$R_k = \begin{bmatrix} 1 & 0\ 0 & 1 \end{bmatrix}.$$

We can now perform the steps to apply joint Kalman filter:

a). Prediction stage: *We use the state transition matrix F_k to predict the state vector at time $k = 1$ based on the state vector and process noise covariance matrix at time $k = 0$:*

$$\begin{aligned} F_k &= \begin{bmatrix} 1 & 0\ 0 & 1 \end{bmatrix}, \\ x_{1|0} &= F_k x_{0|0} = \begin{bmatrix} -75 & 15 \end{bmatrix}, \\ P_{1|0} &= F_k P_{0|0} F_k^T + Q_k = \begin{bmatrix} 25.01 & 0 & 0 & 25.01 \end{bmatrix}. \end{aligned}$$

At the time $k = 2$, we use the state transition matrix F_k to predict state vector at

time $k = 2$ based on the state vector and process noise covariance matrix at time $k = 1$:

$$x_{2|1} = F_k x_{1|1} = \begin{bmatrix} -65 & 22 \end{bmatrix},$$
$$P_{2|1} = F_k P_{1|1} F_k^T + Q_k = \begin{bmatrix} 25.01 & 0\ 0 & 25.01 \end{bmatrix}.$$

b). Update stage: *We use the measurement matrix H_k and measurement noise covariance matrix R_k to update our estimate of the state vector based on the RSSI and SNR measurements at time $k = 1$:*

$$H_k = \begin{bmatrix} 1 & 0 & 0 & 1 \end{bmatrix},$$
$$K_k = P_{1|0} H_k^T (H_k P_{1|0} H_k^T + R_k)^{-1},$$
$$= \begin{bmatrix} 0.9615 & 0\ 0 & 0.961 \end{bmatrix}.$$

We repeat this process for the remaining time steps. In this example, both RSSI (Received Signal Strength Indicator) and SNR (Signal-to-Noise Ratio) are expressed in decibels (dB). It is because decibels are a commonly used unit for quantifying signal strength and ratio measurements in various fields, including telecommunications and signal processing. By utilizing decibels, we can conveniently represent the relative power levels and ratios in a logarithmic scale, which allows for easier comparison and interpretation of the values. The logarithmic nature of decibels allows us to express a wide range of values within a more comprehensible range.

4.3.5 Generating data stream event using sliding windows

In a LoRa network, k different types of interference are created and are denoted as $\{\mathbf{a_1}, \mathbf{a_2}, \cdots, \mathbf{a_k}\}$. Whenever a sensor detects an event, the system generates a data stream event sequence. The system segments the event sequence into fixed-size event count-based windows, each containing an equal number of events. The length of a window is denoted by l, and the event stream and window vector are denoted by $\mathbf{e} = \{e_1, e_2, \cdots, e_i, \cdots\}^T$ and $\mathbf{W} = \{W_1, W_2, \cdots, W_j, \cdots\}^T$, respectively. A window W_j consists of events $\{e_j, e_{j+1}, \cdots, e_{j+l-1}\}$, where e_j and e_{j+l-1} are the first and last events of W_j, respectively. The length of a window l is empirically derived by observing the effect of different values of l on the system's performance, where $l \in [l_{min}, l_{max}]$. The minimum window length l_{min} is calculated as follows: $l_{min} = \min\{\Delta E_1, \cdots, \Delta E_i, \cdots, \Delta E_n\}$, where ΔE_i corresponds to the mean window size of sensor data sequence $\mathbf{a}_i$. The maximum window length l_{max} is the median of the number of events that occur for the

sensory data generated by all LoRa networks: $l_{max} = median\{\Delta E_1, \cdots, \Delta E_n\}$.

Based on experimental results, a window size of 90 for previous events is found to be a good choice. Let's say we have a LoRa network with 3 types of interference, denoted as $\mathbf{a}_1$, $\mathbf{a}_2$, and $\mathbf{a}_3$. Each type of interference generates a data stream event sequence of different lengths:

- $\mathbf{a}_1$ generates event sequences of length 1000.
- $\mathbf{a}_2$ generates event sequences of length 800.
- $\mathbf{a}_3$ generates event sequences of length 1200.

We want to segment each event sequence into fixed-size event count-based windows, with the length of each window denoted as l. We can choose l to be any value between the minimum and maximum window length:

- l_{min} is the minimum value of ΔE for all three types of interference: $l_{min} = \min(1000, 800, 1200) = 800$
- l_{max} is the median value of ΔE for all three types of interference: $l_{max} = \text{median}(1000, 800, 1200) = 1000$

So we can choose any value of l between 800 and 1000. Let's say we choose $l = 900$. Then we can divide each event sequence into windows of length 900:

- $\mathbf{a}_1$ will have $\lceil \frac{1000}{900} \rceil = 2$ windows.
- $\mathbf{a}_2$ will have $\lceil \frac{800}{900} \rceil = 1$ window.
- $\mathbf{a}_3$ will have $\lceil \frac{1200}{900} \rceil = 2$ windows.

In each window, we will have an equal number of events (in this case, 900). We can then analyze each window to detect events of interest and filter out interference.

4.3.6 Identify the interference devices

This section covers the mechanism of identifying the interfering devices, which is a multi-class classification problem. It comprises feature extraction and classification steps. The system extracts the features from the preprocessed data and uses them to identify the interference LNs. We use different features, as presented in Table 4.2.

During the determination of interfering devices, we identify the best-performing classifier for predicting relapses in a high-ceiling smart building. To accomplish this goal, we evaluate the performance of four popular classifiers: **logistic regression**, **linear SVM**, **RBF SVM**, and **random forest**. To avoid over-fitting, we apply elastic net regularization to logistic regression and linear SVM, which linearly combines L1 and L2 penalties. We also use a grid search to identify the best hyper-parameters for each model. To classify a data stream event sequence of **RSSI** and **SNR** in a high-ceiling smart building, we propose a classification function that combines the *SVM*

Table 4.2: Time and frequency domain features considered in this work during implementation.

(a) Time Domain Features

Measure	**Formula**
Mean value	$\mu_x = \frac{1}{n}\sum_{i=1}^{n} f_x(i), \mu_y, \mu_z$
Standard Deviation	$\sigma_x = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f_X(i)-\mu)^2}, \sigma_y, \sigma_z$
Minimum value	$m_x = \{Min(f_x(i))\|1 \leq i \leq n\}, m_y, m_z$
Maximum value	$M_x\{Max(f_x(i))\|1 \leq i \leq n\}, M_y, M_z$
Median value	Med_x = $((n+1)/2)^{th}$value if n is odd, else $(n/2)^{th}$, Med_y, Med_z
Correlation Coefficient	$C_{xy} = \frac{\sum_{i=1}^{n}(f_x(i)-\mu_x)(f_y(i)-\mu_y)}{n\sigma_1\sigma_2}, C_{yz}, C_{zx}$
Angle	θ_x, θ_y, and θ_z
Average energy	$E_x = \sum_{i=1}^{n} f_x(i)^2, E_y, E_z$

(b) Frequency Domain Features

FFT coefficients	$F_x(k) = \sum_{m=1}^{n} f_x(m)e^{-i2\pi km/n}, 1 \leq k \leq n$
Maximum Frequency	$Mf_x = \{Max(F_x(i)\|i = 1 \text{ to } n\}, Mf_y, Mf_z$
Centroid Frequency	$Cf_x = \frac{\sum_{i=1}^{N} i\|F_x(i)\|}{\sum_{i=1}^{n}\|F_x(i)\|}, Cf_y, Cf_z$

loss function, *RSSI propagation module*, and *hand-crafted features*. We optimize this function using gradient descent or other numerical optimization methods to find the optimal values of the weight vector, bias term, and Lagrange multipliers that minimize the loss function and maximize the classification accuracy.

The SVM loss function is defined in terms of minimizing the following equation subject to the given constraints:

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_i \xi_i, \tag{4.9}$$

where $y_i(w^T h_i + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$.

Further, we use an RSSI propagation module to model the effects of signal propa-

gation in a high-ceiling smart building. This module takes into account the distance between *two sensors* and *the corresponding path loss factor* to model the received signal strength. The module's equation is:

$$RSSI = P_t + G(d) - 10n \log_{10}(d) + X, \tag{4.10}$$

where, P_t is transmitted power, n is path loss exponent, and X is a random variable for additive white Gaussian noise.

We derive a feature vector h from the sequence of RSSI and SNR measurements x using handcrafted feature engineering. The feature vector consists of statistical features such as mean, variance, and autocorrelation, which capture different aspects of the raw data. An example of a handcrafted feature vector with ten features is:

$$h = [f_1(x), f_2(x), ..., f_{10}(x)], \tag{4.11}$$

where, $f_1(x)$ is the mean of RSSI measurements, $f_2(x)$ is the standard deviation of RSSI measurements, $f_3(x)$ is the mean of SNR measurements, $f_4(x)$ is the standard deviation of SNR measurements, $f_5(x)$ is the correlation between RSSI and SNR measurements, $f_6(x)$ is the mean of the absolute differences, and so on. Finally, we define the overall classification function as:

$$f(x) = \text{sign}\left(w^T \left(f(W_1 h + b_1) \odot G(d)\right) + b\right), \tag{4.12}$$

where, W_1 and b_1 are the weights and biases of the high-level feature extraction function, $\odot$ is the element-wise multiplication operator, and $G(d)$ is the path loss factor derived from the RSSI propagation module.

The joint classification function of the multiclass classifier, path loss module, and high-level features can be written as follows using Karush-Kuhn-Tucker (KKT) condi-

tions and Lagrange multipliers:

$$\min_{w,b,\xi,\alpha} \frac{1}{2}w^T w + C\sum_i \xi_i - \sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\alpha_{ik}, \tag{4.13a}$$

$$\text{s.t. } y_{ik}(w_k^T h_i + b_k) \geq 1 - \xi_i, \tag{4.13b}$$

$$\xi_i \geq 0, \tag{4.13c}$$

$$\sum_{k=1}^{K} y_{ik} = 1, \tag{4.13d}$$

$$\alpha_{ik} \geq 0, \tag{4.13e}$$

$$\sum_{k=1}^{K} \alpha_{ik} = 1, \tag{4.13f}$$

$$i = 1, \cdots, N,\ k = 1, \cdots, K,$$

where, w_k and b_k are the weight vector and bias term for class k, h_i is the high-level feature vector for data sample i, y_{ik} is the target variable which takes the value 1 if data sample i belongs to class k and 0 otherwise. ξ_i is the slack variable, C is the penalty parameter, and α_{ik} is the Lagrange multiplier.

The path loss module is incorporated into the high-level feature vector h_i as follows:

$$h_i = [f_1(x_i), \cdots, f_{10}(x_i), g_1(d_i), \cdots, g_{10}(d_i)], \tag{4.14}$$

where, $f_j(x_i)$ is the j-th statistical feature of the raw data, as defined previously, and $g_j(d_i)$ is the j-th path loss feature derived from the path loss module based on the distance d_i between the sensor and the access point.

Finally, the joint classification function can be written as:

$$f(x) = \arg\max_k \left(\sum_{j=1}^{K} \alpha_{ij} y_{ij} (w_j^T h_i + b_j) \odot G(d_i) \right), \tag{4.15}$$

where $\odot$ is the element-wise multiplication operator, and $G(d_i)$ is the path loss factor derived from the path loss module based on the distance d_i between the sensor and the access point. The function $f(x)$ returns the class with the highest score among all classes $k = \{1, \cdots, K\}$. The complete process is shown in Algorithm 4.1.

Example 2 *Let us assume, we have a training dataset with* $N = 100$ *samples, each sample having* 2 *features* ***RSSI*** *and* ***SNR*** *and belonging to one of the* 3 *classes. We set the penalty parameter* C *to be* 1 *and the learning rate* η *to be* 0.01*. Next, we can compute*

the high-level feature vectors for all data samples in ***Step 1*** *using the function* $f(x)$*, which could be a neural network. For example, let us say the output of this function for each sample is a vector of length* 10*. In* ***Step 2****, we compute the path loss factor for all data samples using the function* $G(d)$*. Let us assume the output of this function for each sample is a scalar value between* 0 *and* 1*.*

Further, ***Step 3*** *initializes the weight vector* $\boldsymbol{w_k}$*, bias term* b_k*, slack variable* ξ_i*, and Lagrange multiplier* α_{ik} *for each class* $k = 1, \cdots, K$*, data sample* $i = 1, \cdots, N$*. Let us assume* $K = 3$*, so we need to initialize* 3 *weight vectors and bias terms, and for each sample, we need to initialize* 3 *slack variables and* 3 *Lagrange multipliers. Assuming each high-level feature vector is a column vector of length* 10*, the weight vector and bias term for each class can be initialized as:*

$$\boldsymbol{w}_1 = \begin{bmatrix} 0.1 \\ 0.2 \\ \vdots \\ 0.9 \end{bmatrix}, \qquad \boldsymbol{w}_2 = \begin{bmatrix} -0.1 \\ 0.3 \\ \vdots \\ -0.9 \end{bmatrix}, \qquad \boldsymbol{w}_3 = \begin{bmatrix} 0.3 \\ -0.2 \\ \vdots \\ 0.1 \end{bmatrix}.$$

$$b_1 = 0.5 \qquad b_2 = 0.2 \qquad b_3 = -0.3.$$

For each sample, we can initialize the slack variables and Lagrange multipliers to be 0*. In* ***Step 4****, we can start training the model using the gradient descent algorithm. We compute the gradient of the objective function with respect to the parameters and update them using the learning rate until convergence. The objective function involves maximizing the margin between the decision boundary and the closest data points of each class while minimizing the misclassification error. The update equations involve the Lagrange multipliers and slack variables to enforce the Karush-Kuhn-Tucker (KKT) conditions. Once the model has converged, we can use the joint classification function in* ***Step 5*** *to predict the class of a new data sample. We compute the score for each class using the weight vector, bias term, and Lagrange multipliers and select the class with the highest score. The path loss factor is also taken into account by multiplying the score with the path loss factor for the new data sample.*

4.4 Experimental results

This section presents the evaluation of the proposed approach to SQM dataset. Moreover, this work carried out several experiments to answer the following questions:

- What is the impact of the number of data samples on the SNR values in the LoRa

network? (Section 4.4.1)
- How to study the impact of the number of LNs on SNR values in the LoRa network? (Section 4.4.2)
- How to determine the impact of different parameters on the performance of the proposed approach? (Section 4.4.3)
- What is the impact of increasing LGs count on the inference? (Section 4.4.4)
- What is the impact of using different ML models on performance? (Section 4.4.5)
- How different ML models are impacted by the change in the number of LNs? (Section 4.4.6)

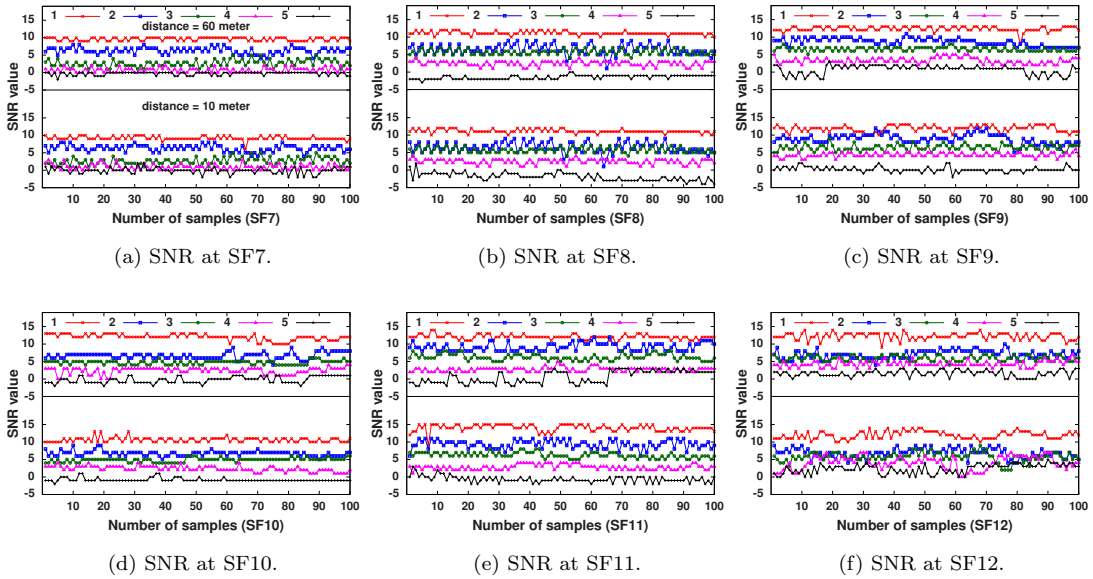
4.4.1 Impact of number of samples on SNR values with varying distance among LNs and SF

The objective of this experiment is to investigate the impact of the number of data samples on SNR values. The experiment involved considering two distances between the LNs and LG, which were 10 and 60 meters, and using the SQM dataset and SVM model. Table 5.3 showed that the SNR values exhibited fluctuating behavior with minimal variance at different numbers of LNs and SF. This was due to the fixed data rate for the LoRa network as the number of data samples increased. The results showed that the SNR values changed from $9 - 11$ to $12 - 15$ for one LN and SF7 to SF12, respectively. Further, with an increase in the number of LNs, there was a higher chance of interference, leading to a reduction in SNR values from LNs count 1 to 5.

In conclusion, this result highlights the importance of considering the number of samples in determining the SNR values. Additionally, the results demonstrate that an increase in the number of LNs can result in a reduction in SNR values due to the possibility of interference. Overall, these findings have implications for the design and implementation of LoRa networks, particularly in situations where SNR values are critical to network performance.

4.4.2 Impact of number of LNs on SNR

The experiment aimed to investigate how the number of LNs affects the SNR values of LNs in a LoRa network. Figure 4.5 shows the SNR fluctuations on different SFs at various combinations of LNs. In a one-way communication scenario, LNs send data to the LG, which calculates the SNR and RSSI values of the received data. LG sends data with RSSI and SNR values to the LoRa network server, which makes a window size of 20 SNR as an input vector for machine learning models. The results reveal



(a) SNR at SF7. (b) SNR at SF8. (c) SNR at SF9.

(d) SNR at SF10. (e) SNR at SF11. (f) SNR at SF12.

Figure 4.4: Average SNR value *vs.* the number of samples using simultaneous communication at different SFs at different distances between LNs and LG

that an increase in the number of LNs leads to higher interference and simultaneous message transmission, resulting in higher SNR fluctuation and lower SNR values of the received signal. Additionally, the LoRa network suffers from the capture effect, where LG gets only the strongest signal among multiple signals approaching it. Figure 4.5 also highlights the high variation in the SNR values, where the pattern of SNR is inconsistent for any number of LNs due to the increasing noise at higher SF. These findings provide insights into the impact of the number of data samples on the SNR values of LNs in a LoRa network, which is crucial for optimizing the network's performance.

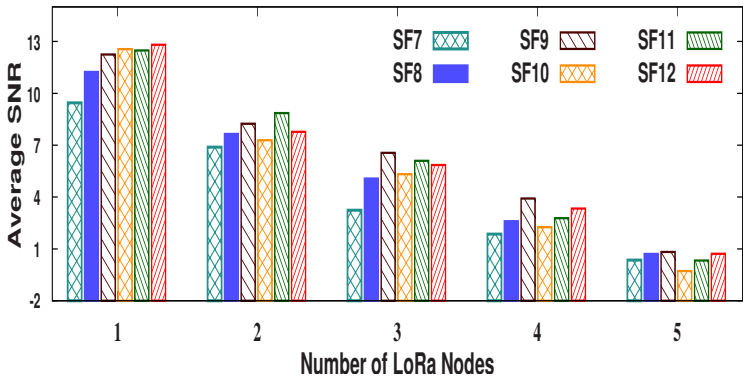


Figure 4.5: Average SNR value on different SFs (SF7 to SF12) with multiple combinations of LNs (1 to 5).

4.4.3 Impact of different parameters on performance

The experiment aims to assess how various parameters, namely Obstacle (with or without a wall), LN, LG, and payload message Size (PS), impact the accuracy and SNR values. Table 4.3 presents the observations collected from the experiment. The study examines the different positions of LNs in collecting data for combinations, including various data sizes and locations, with and without walls. Four distinct scenarios were used to collect datasets. The first scenario involved deploying LNs and LG at a distance of 10 meters without any walls. The second scenario involved placing a concrete wall between LNs and LG, both at a distance of 10 meters. The third scenario involved deploying LNs and LG at a distance of 60 meters without any walls. Lastly, the fourth scenario involved placing a wall between LNs and LG, both at a distance of 60 meters. Table 4.3 illustrates all these scenarios for different SFs (SF7 to SF12).

The following discussion delves into various parameters on accuracy and SNR:

- **Impact of the obstacles:** To assess the impact of obstructions between the LNs and LG, we have specifically chosen concrete walls as the barriers for our analysis. We conducted data collection under two scenarios: with walls (labeled "w") and without walls (labeled "wo"). Our findings indicate that the presence of obstacles compromised the accuracy of our model but did not affect the SNR, as indicated in Table 4.3. This is because the obstructions slowed down the movement of signals, causing interference on the server. However, radio signals were able to penetrate the concrete walls easily, and therefore the SNR values remained unaffected, regardless of whether walls were present during data collection.
- **Impact of LNs with and without obstacles:** We used the same range of LNs counts as in previous experiments, ranging from 1 to 5 (Table 4.3). Our results show that the accuracy of the collected dataset was unaffected by the varying LN counts. However, we observed a significant variance in the SNR values across all SF (i.e., SF7 to SF12), ranging from $9 - 10$ to $-1 - 3$. This variance was due to the increased interference among the LNs, which negatively impacted the SNR value on the LG. We made similar observations in other experimental scenarios, such as varying payload size and the presence of obstacles.
- **Impact of LG with and without obstacles:** During the experimental evaluation, we focused solely on single gateways as increasing the number of LG counts leads to less interference. We will also delve into the impact of LoRa gateways in greater detail later on.
- **Payload message Size:** Payload message size is a critical factor in data transmission from LNs to LGs. Our experiment involves two payload sizes: 10 and 60

bytes. Table 4.3 shows a marginal compromise in accuracy when increasing the payload size due to the longer duration that the allocated radio channel is locked and increased chance of inference. This makes it difficult to estimate interference accurately and can compromise accuracy.

- **Spreading Factor:** Finally, we investigated the impact of spreading factors from SF7 to SF12 on the accuracy and SNR in the scenario of collecting the SQM dataset. We found that increasing the spreading factors led to an increase in the SNR value, but only a marginal fluctuation in accuracy. For instance, when using SF7 and one LN, the range of SNR was between $9-10$, while for SF12, the SNR range was between $11-14$. This is likely due to the fact that higher SFs provide stronger signal strength than lower SFs.

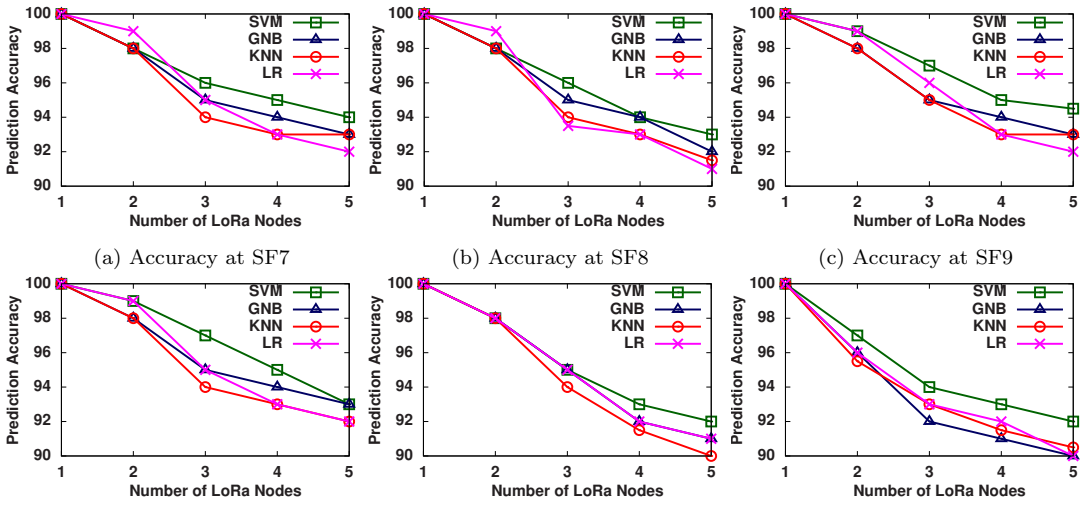
4.4.4 Impact of number of gateways

The experiment investigates how the performance of our proposed approach is affected by the number of gateways. To conduct the experiment, we sent 100 messages from each of the five LNs and five LGs, varying the number of LNs and LGs to observe the impact of different gateway configurations. Our results show that increasing the number of gateways leads to a reduction in interference and message loss. We have presented our findings in Table 4.4, which highlights the impact of the number of gateways on the varying count of LNs. We observed that when the number of LGs matches the number of LNs, there is no interference as direct communication is possible between the LG and LN. However, if a large number of LNs are associated with a single LG, the interference fraction is higher. When only one LG is present, it receives the message with the strongest received signal strength index, while other signals undergo interference in the LoRa network, resulting in a decrease in SNR when messages are sent simultaneously. To mitigate these issues, the gateway in the developed system sends a message containing payload, SNR, and RSSI to the server for analysis. The server then applies machine learning techniques to predict the number of interference LNs and schedules the interference node accordingly. This approach reduces the number of message losses in the LoRa network. In summary, our experiment highlights the importance of having multiple gateways in a LoRa network to minimize interference and message loss. Additionally, our proposed system offers an effective solution for managing interference in LoRa networks using machine learning techniques.

4.4.5 Analyzing the impact of multi-class classification ML models

Our study focuses on evaluating the impact of various multi-class classification machine learning models, namely Gaussian Naive Bayes (GNB), k-Nearest Neighbor (KNN), Logistic Regression (LR), and Support Vector Machine (SVM), on the performance of our proposed approach. We apply these models to analyze the collected SNR pattern and predict the number of interference nodes in the LoRa network. The results presented in Table 4.5 demonstrate that SVM outperforms the other classifiers, achieving the highest accuracy for interference prediction in the LoRa network. This can be attributed to SVM's ability to build the most differentiable hyper-plane on the SQM dataset. Notably, our experiment employs data from three LNs.

Although all other machine learning models deliver comparable accuracy, SVM is more robust against underfitting and overfitting. We observe that as the number of LN increases in the network, the SNR value of the received message decreases, causing interference between simultaneous message transmissions. However, the accuracy of the different machine learning models does not change significantly. Regardless of the scenario, we always achieve high accuracy, exceeding 90%.



(a) Accuracy at SF7 (b) Accuracy at SF8 (c) Accuracy at SF9

(d) Accuracy at SF9 (e) Accuracy at SF10 (f) Accuracy at SF12

Figure 4.6: Illustration of impact of different machine learning models on the varying count of LoRa Nodes (1 to 5).

4.4.6 Impact of increasing LN on ML models performance

In this study, we build upon our previous experiment by examining the effect of increasing the number of LNs on the performance of various learning models discussed in

Table 4.5. Figure 4.6 shows that the impact of changing SFs on the models' performance is minimal, while the impact of the number of LNs is significant. Our findings suggest that as the number of LNs increases, the accuracy of the models decreases due to a higher chance of interference. This reduces the ability to accurately identify the number of interfering signals, ultimately leading to a decrease in the proposed approach's performance, as depicted in Figure 4.6.

4.4.7 Energy efficiency and Cost-effectiveness

The evaluation parameters for defining energy efficiency and cost-effectiveness in the proposed approach are as follows:

1. **Energy Efficiency:**
   - *Low Power Consumption:* The approach uses LoRaWAN, known for its low power consumption during long-range communication.
   - *Battery Maintenance:* Sensors with long battery life requiring infrequent replacements or recharging are considered energy-efficient.
   - *Signal-to-Noise Ratio (SNR):* Higher SNR values indicate better communication quality and efficiency, which is critical for reducing energy usage by minimizing retransmissions.
2. **Cost-Effectiveness:**
   - *Sensor Placement Strategy:* Efficient placement of sensors in high-ceiling buildings minimizes costs related to installation and maintenance.
   - *Maintenance Costs:* Evaluating the downtime and maintenance effort required for faulty sensors provides insights into cost-effectiveness.
   - *Hardware Independence:* The proposed machine-learning model is designed to work with existing hardware, avoiding the need for additional cost-intensive setups.
   - *Dataset Utilization:* Openly available datasets reduce development costs and facilitate cost-effective research and application deployment.

These parameters collectively ensure that the proposed approach achieves energy efficiency and cost-effectiveness while maintaining robust performance under real-world constraints.

4.5 Conclusion and future work

In this chapter,we present a novel approach for identifying interference nodes in LoRa networks. Our approach utilizes machine learning techniques, using SNR and RSSI values to accurately identify nodes that cause network interference. We conducted experiments in different deployment scenarios, including a high-ceiling smart building, and observed that network parameter-based interference solutions work well in scenarios with few obstacles and static LNs and LG. Our approach successfully estimates interference from up to five devices on a given SF with high accuracy. We also observed that interference among LNs with low SF is higher, likely due to the proximity of all devices to each other and the LG.

In future work, we plan to allocate optimal SF and transmission power to LNs based on the number of interference LNs detected, optimizing the energy consumption of the LoRa network. We also plan to explore scenarios where LNs and LG can change position, expanding the applicability of our approach.

Algorithm 4.1: Training a multi-class classifier model.

Input: Training data X and labels y, high-level feature function $f(x)$, path loss module $G(d)$, penalty parameter C;

for $i \leftarrow 1$ *to* N **do**
  Compute the high-level feature vectors for all data samples: $h_i = f(x_i)$;
  Compute the path loss factors for all data samples: $p_i = G(d_i)$;
  for $k \leftarrow 1$ *to* K **do**
    Initialize vector w_k, bias term b_k, slack variable ξ_i, and Lagrange α_{ik};
    Set the learning rate η;
Train the model using the following steps until convergence;
do
  I. Compute the gradient of the objective function with respect to the parameters:

$$\nabla_w L = w - \sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\alpha_{ik}h_i, \nabla_b L = -\sum_{i=1}^{N}\sum_{k=1}^{K} y_{ik}\alpha_{ik}, \nabla_{\xi_i} L = C - \sum_{k=1}^{K}\alpha_{ik},$$
$$\nabla_{\alpha_{ik}} L = y_{ik}(w_k^T h_i + b_k) - 1 + \xi_i.$$

  II. Update the parameters using the learning rate:

$$w_k \leftarrow w_k - \eta\nabla_w L, b_k \leftarrow b_k - \eta\nabla_b L, \xi_i \leftarrow \xi_i - \eta\nabla_{\xi_i} L, \alpha_{ik} \leftarrow \alpha_{ik} - \eta\nabla_{\alpha_{ik}} L.$$

  III. Enforce the KKT conditions:

$$\alpha_{ik} = 0 \text{ if } y_{ik}(w_k^T h_i + b_k) \geq 1 - \xi_i, \alpha_{ik} = C \text{ if } y_{ik}(w_k^T h_i + b_k) \leq 1 - \xi_i, 0 \leq \alpha_{ik} \leq C.$$

while *convergence*;
Joint classification function can be used to predict the class of a new data sample:

$$f(x) = \arg\max_k \Big(\sum_{j=1}^{K} \alpha_{ij} y_{ij} (w_j^T h_i + b_j) \odot p_i \Big). \tag{4.16}$$

return Trained multi-class classifier model;

Table 4.3: Impact of different parameters on accuracy (in %) of SVM to predict interfering LNs in LoRa network on SQM dataset. Acc = Accuracy, Obs= Obstacle (with or without wall denoted as *w* and *wo*, respectively), LN = LoRa Nodes, LG = LoRa Gateway, PS = Payload message Size, SNR =Singnal-to-Noise Ratio.

SF	LN	LG	PS	Obs	SNR	Acc	LN	LG	PS	Obs	SNR	Acc	LN	LG	PS	Obs	SNR	Acc	LN	LG	PS	Obs	SNR	Acc
	1	1	10	*wo*	9 – 10		1	1	10	*w*	9 – 10		1	1	60	*wo*	9 – 10		1	1	60	*w*	9 – 10	
	2	1	10	*wo*	7 – 9		2	1	10	*w*	7 – 9		2	1	60	*wo*	7 – 9		2	1	60	*w*	7 – 9	
SF7	3	1	10	*wo*	4 – 7	96	3	1	10	*w*	4 – 7	94	3	1	60	*wo*	4 – 7	93	3	1	60	*w*	4 – 7	91
	4	1	10	*wo*	2 – 5		4	1	10	*w*	2 – 5		4	1	60	*wo*	2 – 5		4	1	60	*w*	2 – 5	
	5	1	10	*wo*	−1 – 3		5	1	10	*w*	−1 – 3		5	1	60	*wo*	−1 – 3		5	1	60	*w*	−1 – 3	
	1	1	10	*wo*	10 – 12		1	1	10	*w*	10 – 12		1	1	60	*wo*	10 – 12		1	1	60	*w*	10 – 12	
	2	1	10	*wo*	6 – 9		2	1	10	*w*	6 – 9		2	1	60	*wo*	6 – 9		2	1	60	*w*	6 – 9	
SF8	3	1	10	*wo*	4 – 7	96	3	1	10	*w*	4 – 7	94	3	1	60	*wo*	4 – 7	93	3	1	60	*w*	4 – 7	92
	4	1	10	*wo*	2 – 4		4	1	10	*w*	2 – 4		4	1	60	*wo*	2 – 4		4	1	60	*w*	2 – 4	
	5	1	10	*wo*	−2 – 3		5	1	10	*w*	−2 – 3		5	1	60	*wo*	−2 – 3		5	1	60	*w*	−2 – 3	
	1	1	10	*wo*	10 – 13		1	1	10	*w*	10 – 13		1	1	60	*wo*	10 – 13		1	1	60	*w*	10 – 13	
	2	1	10	*wo*	7 – 10		2	1	10	*w*	7 – 10		2	1	60	*wo*	7 – 10		2	1	60	*w*	7 – 10	
SF9	3	1	10	*wo*	5 – 8	96	3	1	10	*w*	5 – 8	94	3	1	60	*wo*	5 – 8	93	3	1	60	*w*	5 – 8	92
	4	1	10	*wo*	3 – 5		4	1	10	*w*	3 – 5		4	1	60	*wo*	3 – 5		4	1	60	*w*	3 – 5	
	5	1	10	*wo*	−1 – 3		5	1	10	*w*	−1 – 3		5	1	60	*wo*	−1 – 3		5	1	60	*w*	−1 – 3	
	1	1	10	*wo*	10 – 13		1	1	10	*w*	10 – 13		1	1	60	*wo*	10 – 13		1	1	60	*w*	10 – 13	
	2	1	10	*wo*	7 – 10		2	1	10	*w*	7 – 10		2	1	60	*wo*	7 – 10		2	1	60	*w*	7 – 10	
SF10	3	1	10	*wo*	4 – 7	95	3	1	10	*w*	4 – 7	94	3	1	60	*wo*	4 – 7	93	3	1	60	*w*	4 – 7	91
	4	1	10	*wo*	1 – 4		4	1	10	*w*	1 – 4		4	1	60	*wo*	1 – 4		4	1	60	*w*	1 – 4	
	5	1	10	*wo*	−1 – 2		5	1	10	*w*	−1 – 2		5	1	60	*wo*	−1 – 2		5	1	60	*w*	−1 – 2	
	1	1	10	*wo*	11 – 14		1	1	10	*w*	11 – 14		1	1	60	*wo*	11 – 14		1	1	60	*w*	11 – 14	
	2	1	10	*wo*	7 – 11		2	1	10	*w*	7 – 11		2	1	60	*wo*	7 – 11		2	1	60	*w*	7 – 11	
SF11	3	1	10	*wo*	5 – 8	95	3	1	10	*w*	5 – 8	94	3	1	60	*wo*	5 – 8	93	3	1	60	*w*	5 – 8	91
	4	1	10	*wo*	2 – 5		4	1	10	*w*	2 – 5		4	1	60	*wo*	2 – 5		4	1	60	*w*	2 – 5	
	5	1	10	*wo*	−1 – 4		5	1	10	*w*	−1 – 4		5	1	60	*wo*	−1 – 4		5	1	60	*w*	−1 – 4	
	1	1	10	*wo*	11 – 14		1	1	10	*w*	11 – 14		1	1	60	*wo*	11 – 14		1	1	60	*w*	11 – 14	
	2	1	10	*wo*	6 – 9		2	1	10	*w*	6 – 9		2	1	60	*wo*	6 – 9		2	1	60	*w*	6 – 9	
SF12	3	1	10	*wo*	5 – 7	96	3	1	10	*w*	5 – 7	94	3	1	60	*wo*	5 – 7	92	3	1	60	*w*	5 – 7	91
	4	1	10	*wo*	2 – 5		4	1	10	*w*	2 – 5		4	1	60	*wo*	2 – 5		4	1	60	*w*	2 – 5	
	5	1	10	*wo*	0 – 4		5	1	10	*w*	0 – 4		5	1	60	*wo*	0 – 4		5	1	60	*w*	0 – 4	

Table 4.4: Illustration of impact of the number of gateways on inference and fraction of message loss. LN = LoRa Node and LG = LoRa Gateway.

Configuration	**Spreading factor**						**Interference**
	SF7	*SF8*	*SF9*	*SF10*	*SF11*	*SF12*	
1-*LN*, 1-*LG*	0	0	0	0	0	0	No
2-*LNs*, 1-*LG*	3/100	3/100	4/100	5/100	5/100	6/100	Yes
2-*LNs*, 2-*LGs*	0	0	0	0	0	0	No
3-*LNs*, 1-*LG*	7/100	8/100	8/100	11/100	11/100	12/100	Yes
3-*LNs*, 2-*LGs*	5/100	6/100	7/100	7/100	7/100	9/100	Yes
3-*LNs*, 3-*LGs*	0	0	0	0	0	0	No
4-*LNs*, 1-*LG*	9/100	9/100	10/100	13/100	14/100	15/100	Yes
4-*LNs*, 2-*LGs*	6/100	6/100	6/100	7/100	9/100	10/100	Yes
4-*LNs*, 3-*LGs*	4/100	5/100	5/100	6/100	6/100	6/100	Yes
4-*LNs*, 4-*LGs*	0	0	0	0	0	0	No
5-*LNs*, 1-*LG*	12/100	13/100	14/100	14/100	15/100	15/100	Yes
5-*LNs*, 2-*LGs*	9/100	10/100	10/100	10/100	11/100	11/100	Yes
5-*LNs*, 3-*LGs*	7/100	8/100	8/100	9/100	9/100	9/100	Yes
5-*LNs*, 4-*LGs*	3/100	4/100	4/100	5/100	5/100	5/100	Yes
5-*LNs*, 5-*LGs*	0	0	0	0	0	0	No

Table 4.5: Accuracy (in %) of different machine learning models, while predicting interfering LNs in LoRa network on SQM dataset.

Machine learning -based model	**Spreading factors**					
	SF7	*SF8*	*SF9*	*SF10*	*SF11*	*SF12*
Support Vector Machine	**95.63**	**95.16**	**96.51**	**96.10**	**95.87**	**95.34**
Gaussian Naive Bayes	94.87	94.10	95.00	95.33	95.10	94.33
k-Nearest Neighbor	94.51	94.12	95.10	94.87	95.33	94.63
Logistic Regression	95.51	95.10	94.86	94.63	94.33	95.12