INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Heterogeneous Communication Scheme for IoT Smart Nodes

Beatriz Carolina Duque Dias

Master in Telecommunications and Computer Engineering

Supervisor
PhD Pedro Joaquim Amaro Sebastião, Assistant Professor,
Iscte - Instituto Universitário de Lisboa

Co-Supervisor
Master André Filipe Xavier da Glória, Researcher,
Iscte - Instituto Universitário de Lisboa

October, 2021

TECNOLOGIAS
E ARQUITETURA

Department of Information Science and Technology

Heterogeneous Communication Scheme for IoT Smart Nodes

Beatriz Carolina Duque Dias

Master in Telecommunications and Computer Engineering

Supervisor
PhD Pedro Joaquim Amaro Sebastião, Assistant Professor,
Iscte - Instituto Universitário de Lisboa

Co-Supervisor
Master André Filipe Xavier da Glória, Researcher,
Iscte - Instituto Universitário de Lisboa

October, 2021

Acknowledgment

First, I would like to thank Professor Pedro Sebastião for the orientation and support given through this dissertation.

To André Glória, my biggest thanks for all the support and guidance given, always available to help me with all the doubts and the problems that appeared during the development of this dissertation. For that, I am truly grateful.

A huge thanks to my family, especially to my parents Francisco and Zélia and my sister Catarina for their unconditional love and support, who have always motivated and encouraged me to achieve my goals, without them none of this would be possible.

Finally a big thanks to all my friends and colleagues who accompanied me on this journey but especially to Carlos and Maria that during these 5 years were always there for me when I needed them.

Resumo

Tem existido uma enorme evolução na área da Internet das Coisas isto faz com que também tenha começado a existir uma maior necessidade de melhorar as suas comunicações para um melhor desempenho dos dispositivos.

Esta dissertação apresenta uma proposta de um sistema de comunicação heterogéneo que seja eficiente e que, com ajuda de algoritmos de Machine Learning será possível prever, automaticamente sem intervenção humana, qual o sistema de comunicação usar. O objetivo deste sistema passa por garantir que o utilizador esteja sempre a usar a tecnologia mais adequada para a situação e ambiente em que se encontra e com isto tornar a experiência do dispositivo a melhor possível. O sistema, que inclui software e hardware, foi desenvolvido de forma a ser fácil de usar e de baixo custo.

Foi desenvolvida também uma plataforma online, em que utilizador consegue visualizar e monitorizar o sistema em tempo real. Na plataforma é possível introduzir os vários campos que são apresentados, isto é, as características do ambiente em que o dispositivo se encontra e depois o utilizador consegue obter as informações acerca da tecnologia mais adequada a usar, para este caso específico.

A eficácia do sistema desenvolvido também foi estudada e testada experimentalmente e foi possível observar uma precisão a rondar os 94% para o edge computing e 96,78% para a API. Os resultados obtidos foram explicados detalhadamente nesta dissertação.

Palavras-Chave: Internet das Coisas, Comunicações Heterogéneas, Protocolos de Comunicação sem Fios, Aprendizagem Automática, Sustentabilidade.

Abstract

There has been a rapid evolution in the area of the Internet of Things, and this means that a greater need for improved communications, for better device performance, is also in need.

This dissertation presents a proposal for an efficient heterogeneous communication system that, with the help of Machine Learning algorithms, will be able to automatically predict, without human intervention, the best communication method to use. The goal of this system is to ensure that the user is always using the most appropriate technology for the situation and environment he is in, and to make the device experience the best possible. The system, which includes software and hardware, was developed to be easy to use and low-cost.

An online platform was also developed, where the user can view and monitor the system in real time. In the platform it is possible to enter the various fields that are presented, that is, the characteristics of the environment in which the device is, and then the user can get information about the best technology to use, for this specific case.

The effectiveness of the system developed was also studied and experimentally tested, and accuracy of between 94% and 96.78% could be observed depending on the scenario. The results obtained were explained in detail in this dissertation.

Keywords: Internet of Thing, Heterogeneous Communications, Wireless Communications Protocols, Machine Learning, Sustainability.

Contents

List of Figures

List of Tables

List of Acronyms

3GPP: 3rd Generation Partnership Project

AP: Access Point

API: Applications Programming Interface

BLE: Bluetooth Low Energy

BSS: Basic Service Set

CC: Cloud Computing

CPU: Central Processing Unit

CSS: Chirp Spread Spectrum

DBPSK : Differential Binary Phase-Shift Keying

DS: Distribution System

DT: Decision Trees

EC: Edge Computing

EPC: Electronic Product Codes

ESS: Extended Service Set

FFD: Full Function Device

GFSK: Gaussian Frequency-Shift Keying

GPIO: General Purpose Input-Output

GSM: Global System for Mobile Communications

I2C: Inter-Integrated Circuit

IBSS: Independent Basic Service

IDE: Integrated Development Environment

IEEE: Institute of Electrical and Electronics Engineers

IoT: Internet of Things

IP: Internet Protocol

ISM: Industrial, Scientific and Medical

LoRa: Long Range

LPWAN: Low Power Wide Area Network

LTE: Long Term Evolution

MAE: Mean Absolute Error

ML: Machine Learning

MLP: Multi-layer Perceptron

NB-IoT: Narrowband Internet of Things

NN: Neural Network

OFDM: Orthogonal Frequency-Division Multiplexing

P2P: Peer-to-Peer

PRB: Physical Resource Block

RF: Random Forest

RFD: Reduced Function Device

RSSI: Received Signal Srength Indication

SC-FDMA: Single-Carrier Frequency-Division Multiple Access

SPI: Serial Peripheral Interface

SVM: Support Vector Machine

UART:Universal Asynchronous Receiver-Transmitter

uCode: Ubiquitous Code

UNB: Ultra-Narrow Band

URL: Uniform Resource Locator

Wi-Fi: Wireless Fidelity

WLAN: Wireless Local Area Network

WPAN: Wireless Personal Area Networks

WSN: Wireless Sensor Network

CHAPTER 1

Introduction

1.1. Motivation and Framework

The choice of this theme arises with the exponential growth of connecting everything that surrounds us to the internet. Technology, definitely, takes a major role in our life, we are already so used to it that we can no longer live without it. Nowadays, there are more and more things that are connected to the internet, a simple object can connect to the internet and thus it becomes a smart object. Right now, we already live in a smart world and, without a doubt, it will not stop here, the Internet of Things will continue to grow. For these reasons, IoT users feel that devices need to be efficient when using them, in order to have a better experience.

With all this growth communication systems will be more complex and, certainly in the near future network environments will be extremely heterogeneous. However, network heterogeneity also brings with it enormous challenges, as devices will have to be extremely capable in order to intelligently roam around heterogeneous networks operating under a wide range of protocols.

With the emergence of these problems, it becomes difficult for a sensors network to be able to work in any scenario, because on many occasions there is a lack of coverage or inefficiency of the communication protocols in certain environments, especially when comparing indoor and outdoor environments, or cities and rural areas. For these reasons, it is important to have an intelligent device that has the ability to manage the best communication method to use depending on the conditions. Therefore, with this ability the device not only improves its efficiency but also reduces its consumption, thus contributing to the sustainability of the network and the planet.

1.2. Objectives

This project aims to create a heterogeneous communication scheme capable of using the most common communication protocols in IoT to be connected to an existing Smart Node, that can be applied in any environment.

This communication model will analyse which is the best communication method to use, depending on the scenario or specification, when new data are sent, autonomously and effectively, without human intervention.

A Machine Learning algorithm will test the communication in several environments and based on the conditions indoor/outdoor type, line of sight, number of obstacles, RSSI, location, distance to the receiver predict the best protocol to use.

To achieve this, several Machine Learning approaches need to be studied, in order to understand which creates the best prediction mechanism.

In the end, it is expected to have a platform/script in which it will be possible to enter the data and return the best communication protocol to use.

1.3. Scientific Contribution

This dissertation presents the following contributions:

- Development and implementation of a communication system that can adapt to any scenario and that operates without human intervention;
- Study of several technologies to be used in the implementation of this project;
- Study and train various machine learning algorithms that are capable of predicting the signal quality;
- Development of an online platform where the user can access and monitor the data in real time;
- Demonstration of the implemented system and various tests that were performed.

The work developed and the results obtained in this thesis has also resulted in a paper published in an international telecommunication conference:

- B. Dias, A. Glória and P. Sebastião, "Prediction of Link Quality for IoT Cloud Communications supported by Machine Learning", in IEEE World AI IoT Congress 2021.

1.4. Thesis Structure

This dissertation is composed of six chapters. This chapter describes the introduction to the project, the motivation and the main objectives, then Chapter 2 presents the state of the art, definitions and important concepts for the realization of the project, and finally, it is presented some related work and what this proposed system can improve in comparison to existing projects. Chapter 3 focuses on the architecture of the system and its entire methodology, this chapter presents the implementations that were made, describes the studies that were conducted on the most common protocols and how the data was collected and then implemented in machine learning algorithms for data analysis. The applications developed, the functionalities of the system and its hardware are also presented. Chapter 4 presents the study on Machine Learning, describing the methodology, the study and the comparison of different algorithms that were carried out in order to understand which one achieves the best performance in the proposed system. Chapter 5 presents the implementation of the system, the various tests that were performed and the discussion of the results. Finally, Chapter 6 presents the main conclusions obtained with the development of the system and the future work that can be implemented to improve this project.

CHAPTER 2

State of the Art

This chapter will present the research and resources for the development of this project. In Section 2.1, we start by analysing the IoT concept and its different components. In Section 2.2 we discuss about WSN. In the next section, we will analyse in more detail the most common communication protocols. In Section 2.4 the Machine Learning concept is introduced. Then in Section 2.5, it is introduced the concepts of cloud and edge computing. To finish this chapter in Section 2.6 some related work that has been done related to the subject and how this project will differ is addressed.

2.1. Internet of Things

The Internet, without a doubt, is one of the most important inventions ever, that has changed people's lives in many different ways, from social relations to professional life. IoT is considered part of the Internet of the future [**1**]. IoT refers to the billions of physical devices around the world that are connected to the internet and that have the ability to gather information around them, perform tasks, and communicate with each other. With the success of the IoT, its fields of applications are increasingly diverse [**2**]. The areas of applications that have grown the most include, e.g. wearables, smart industry, smart home and smart farming. In Figure 2.1, it is represented the concept of IoT.

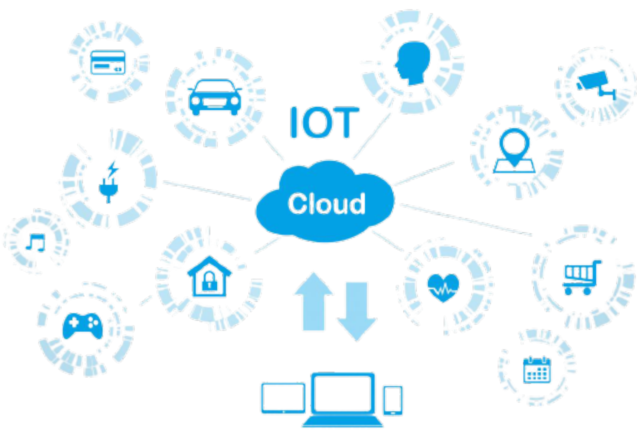

FIGURE 2.1. IoT

IoT can be divided into six blocks, as shown in Figure 2.2, and it is important to understand them because it will help to have a better knowledge of how IoT works, i.e., identification, sensing, communication, computation, services and semantics [**3**].

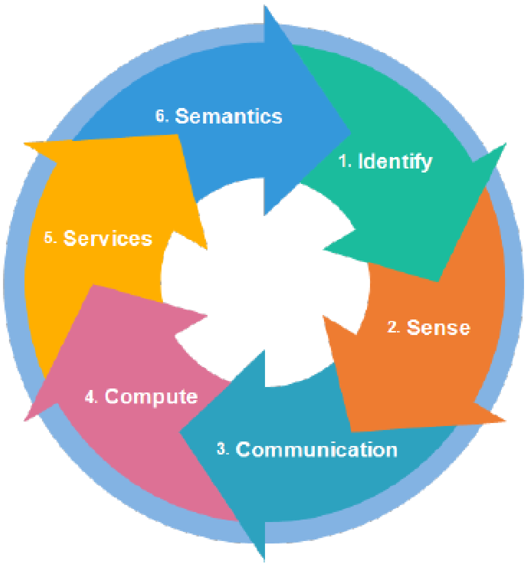

Figure 2.2. IoT Elements

Identification is essential for IoT to name and match services with their demand. It is very important to know the difference between object ID and its address, therefore addressing the IoT object is crucial. There are several identification methods such as electronic product codes (EPC) and ubiquitous codes (uCode).

Sensing consists of gathering data from smart objects within the network and sending it to a data storage unit. Depending on the collected data, these will be analyzed, so that the necessary actions are taken. The IoT sensors can be wearable sensing devices, smart sensors or actuators.

Communication is one of the main goals of IoT, in which different devices are connected to each other and communicate. In communication, devices may send and receive different types of information. There are many communications protocols for the IoT, such as Wi-Fi, Bluetooth, ZigBee and Low-Power Wide-Area Network.

Computation represents the "brain" and the computational ability of the IoT. Computation is the hardware processing units such as microcontrollers, microprocessors or system on chips (SoCs), and software applications perform the task.

Services in IoT can be divided into four categories: identity-related service that is used to get the identity of objects that have sent the request. The information aggregation service purpose to collect all the information from objects and processing is also performed by the aggregation service. Collaborative-aware service, the third service, make decisions according to the collected information and sends appropriate responses to the devices. The last service is a ubiquitous service that offers collaborative-aware services to anyone on demand anytime and anywhere.

Semantics is the ability to extract knowledge intelligently to provide the required services, including discovering and utilizing resources, modeling information, recognizing and analyzing data.

2.2. Wireless Sensor Network

As mention before, one of the key points of IoT is to gather data from a smart object within a sensor network a sending it to a data storage unit. Nowadays the most common is the use of Wireless Sensor Network.

A wireless sensor network (WSN) is constituted by autonomous devices that communicate wirelessly, that are spatially distributed, and detecting certain events of significance in the environmental conditions [**4**], typically used to sense and control some type of events such as temperature, humidity, pressure or sound. Wireless sensor networks consist, essentially, of low-power, light-weight sensor nodes.

WSN has been used extensively in the scientific and technological fields, such as military, agriculture, sports, medicine, and industrial applications [**5**].

Typically, the architecture of a WSN consists of three components: sensor nodes, gateway and observer (user). Gateways and observers are interconnected, most commonly via the internet [**6**]. A typical sensor node consists of a power unit radio, sensing unit and a processing unit [**7**].

Some of the most common topologies found in WSN are P2P (Peer-to-Peer), Star, Tree and Mesh.

2.3. Wireless Communication Protocols

One of the other main components of IoT is communication. It is important to use the appropriate protocol so that the communication is efficient.

In the rapidly growing IoT, applications from personal electronics to industrial machines and sensors are getting wirelessly connected to the Internet.

Nowadays it is possible to reduce or eliminate wired technologies, which are usually expensive. Several new wireless technologies have emerged and resolve some constraints that existed. For example, the quality of service and some constraints related to the application and the environment that surrounds them [**8**].

The most common protocols used in short-range wireless network communications are IEEE 802.11 (Wi-Fi), IEEE 802.15.1 (BLE) and IEEE 802.15.4 (ZigBee). On the other hand, LoRa, SigFox and NB-IoT are the most common Low-power Wide-area Network technologies.

In Table 2.1, [**9, 10, 11, 12**], is presented the main characteristics of each of the wireless communication protocols.

TABLE 2.1. Wireless Communication Protocols Characteristics

Features	**Wi-Fi**	**BLE**	**ZigBee**
IEEE Standard	IEEE 802.11	IEEE 802.15.1	IEEE 802.15.4
Frequency(GHz)	2.4; 5	2.4	2.4
Max signal rate (Mbps)	54	1	0.25
Range (m)	100-150	10	10-100
Nodes	30-250	7	>65000
Power Consumption (mA)	100-350	15	1-10
Complexity	High	Medium	Low

2.3.1. Wi-Fi – IEEE 802.11

Wi-Fi technology is one of the most successful technologies [**13**].

Wi-Fi is based on the IEEE 802.11 standard, provides a wireless connection to a device within a Wireless Local Area Network (WLAN). Over the years the evolution of the standards has shown an increase in data rates, from the 2 Mbit/s IEEE 802.11 to the 11 Mbit/s of 802.11b, the 54 Mbit/s of 802.11a/g, the 600 Mbit/s of 802.11n, and the above Gbit/s rates of the latest 802.11ac [**14**].

The IEEE 802.11 architecture, as shown in Figure 2.3, consists of several components that interact to provide a WLAN that can support station mobility. The fundamental building block of the 802.11 architecture is the cell, known as the basic service set (BSS). A BSS typically contains one or more wireless stations and a central base station, known as an Access Point (AP), and to be able to communicate to each another the stations must be within the BSS range. The stations can also group themselves to form an ad hoc network – a network without an AP, this type of IEEE 802.11 WLAN is often formed without pre-planning, for only as long as WLAN is needed and it is called independent basic service set (IBSS). Also based on the BSS, IEEE 802.11 employs the extended service set (ESS) network configuration. ESS consists of multiple BSSs, the APs of the BSS are connected by a distribution system (DS). The DS with APS allows IEEE 802.11 to create an ESS network of arbitrary size and complexity [**9**]

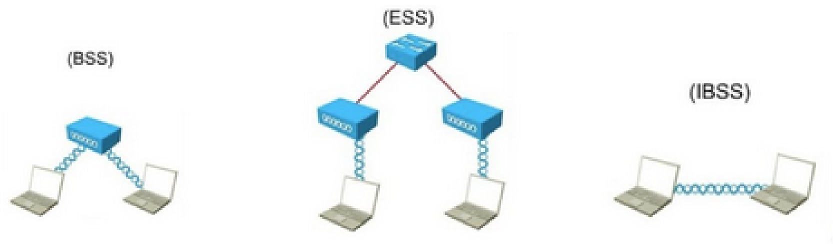

FIGURE 2.3. Wi-Fi Networks Configuration

The Wi-Fi range depends on the version of Wi-Fi that the device is running. As the version evolves, the range increases, for example, IEEE 802.11a has a range of, approximately, 120 meters and IEEE 802.11ac is 250 meters.

Advantages:

- Wi-Fi has a decent range cover and can penetrate walls and other obstacles;
- Adding and removing devices in a Wi-Fi network is a simple process.

Disadvantages:

- High power consumption;
- Radio waves in the network may interfere with other equipment, this can cause connectivity issues or may result in weak signals strength downing transfer speed.

2.3.2. Bluetooth – IEEE 802.15.1 and BLE

Bluetooth, standard IEEE 802.15.1, is a network specification of the Wireless Personal Area Networks (WPAN) and uses the 2.4 GHz frequency. It is based on cheap and short-range wireless radio systems designed to replace cables used in, for example, printers, headphones, mice and keyboards [**15**]. The devices can communicate by the formation of a Bluetooth network, this network has a master unit that controls the other devices, which are designated as slaves. The maximum number of slaves that can simultaneously be active is seven [**16**]. It follows the star network topology, which means that other slaves cannot talk to each other.

Masters and slaves can switch roles in order to participate and be interconnected with other networks. Slaves only communicate with their master in a point-to-point way under the master's control. The master's transmissions may be either point-to-point or point-to-multipoint. To reduce power consumption, the slaves can switch from active mode to parked or standby mode [**9**]. Figure 2.4 shows the Bluetooth topology.

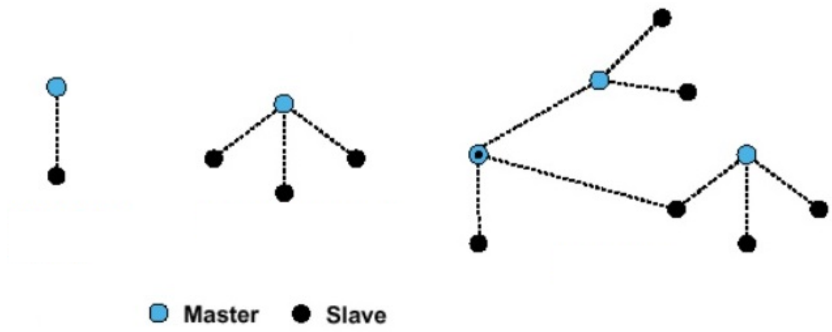

FIGURE 2.4. Bluetooth Topology

There is also Bluetooth low energy (BLE), also known as Bluetooth smart, which is a low-power, low-cost and low-complexity wireless technology [**17**]. The BLE aims to facilitate short-range communication for devices that not required a large amount of data transfer. BLE was designed for low bandwidth and low latency for IoT applications. It has lower power consumption and lower setup time compared to classic Bluetooth [**10, 18**].

Both Bluetooth and BLE technology operate on a frequency of 2.4GHz and employ the same modulation scheme – Gaussian Frequency Shift Keying [**19**].

Advantages:

- Widely supported by mobile phones and tablets, endless devices;
- One coordinator can control a huge amount of slaves.

Disadvantages:

- Only supports star topology;
- A limited number of nodes.

2.3.3. ZigBee – IEEE 802.15.4

ZigBee is a high-level communication protocol, based on low-power wireless IEEE 802.15.4 network standard, it is often used in applications that require a low data rate and longer battery life [**10**].

This standard has two types of devices: Full Function Device (FFD) and Reduced Function Device (RFD). FFD can communicate with RFDs and other FFDs, however, the RFD can only communicate with an FFD. The FFD can perform all the tasks that are defined by the ZigBee standard and can operate in three modes: as a coordinator, as a router and as an end-device [**11**]. The coordinator is the brain of the ZigBee network, it commissions devices of the network, stores the security keys and also bridges to other networks. There is only one Zigbee Coordinator in any network. The router works as an intermediary or to transmit data within the network. The end device can only talk to the parent node (router or coordinator), it cannot talk to other devices directly. Figure 2.5 shows the Zigbee topologies.

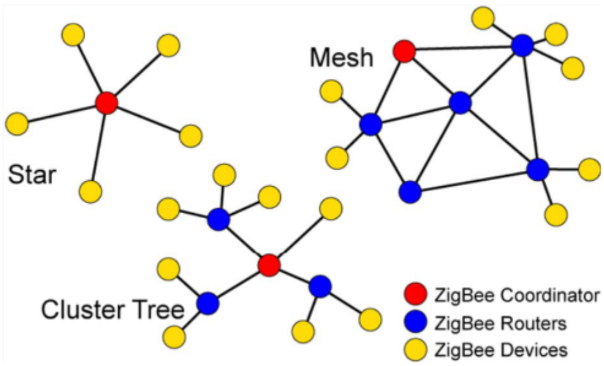

FIGURE 2.5. ZigBee Topologies

On the other hand, RFD has more simple jobs such as reading temperature data. RFDs can only communicate with a coordinator and are limited to a star topology.

Advantages:

- Low power;
- Supports many salves;
- One coordinator can control many slaves.

Disadvantages:

- Requires extra hardware;
- Is incompatible with other network protocols.

2.3.4. ESP-Now

ESP-Now is a peer-to-peer wireless protocol developed by Espressif which enable multiple devices to communicate with one another without using Wi-Fi. The pairing between devices is needed prior to their communication. So, after paring a device with each other, the connection is persistent [**20**]. This means that if suddenly one of the boards loses power or resets, when it restarts, it will automatically connect to its peer to continue the communication. This protocol enables a low power consumption between multiple devices. It more power-efficient and faster to deploy when compared to Wi-Fi [**21**].

Advantages:

- Supports a maximum of 20 nodes;
- No overhead;
- Faster data transmission since it does not need to connect to a Wi-Fi Access Point.

Disadvantages:

- Data packet is limited to 250 bytes;
- Limited to up to 10 encrypted peers.

Table 2.2. Low Power Wide Area (LPWAN) Technologies Characteristics

Features	**NB-IoT**	**LoRa**	**SigFox**
Frequency (MHz)	Licensed LTE frequency bands	868	868
Max Signal Rate (kbps)	20 (UL) 200 (DL)	50	0.1 (UL) 0.6 (DL)
Range (km)	1 (urban) 10 (rural)	5 (urban) 20 (rural)	10 (urban) 50 (rural)
Power Efficiency	Medium High	Very High	Very High

2.3.5. NB-IoT

Narrowband Internet of Things (NB-IoT) is a Low Power Wide Area Network (LPWAN) technology. NB-IoT is a specification developed by 3GPP and was standardized as part of 3GPP Release 13 in June 2016. The NB-IoT is an evolution of the LTE system and operates with a carrier bandwidth of 180 kHz. NB-IoT significantly improves the power consumption of user devices, system capacity and spectrum efficiency, especially in deep coverage. A battery life of more than 10 years can be supported for a wide range of use cases.

NB-IoT can be deployed in three different operation modes [**22**]:

- Standalone: NB-IoT can occupy one GSM channel (200 kHz);
- in-band operation utilizing resource blocks within an LTE carrier;
- guard band operation: utilizing resource blocks within an LTE carrier.

The downlink transmission of NB-IoT is similar to that of LTE in the time domain with 10 ms length. Each frame consists of 10 subframes of 1 ms length and each subframe has 2 slots with a length of seven OFDM symbols. In the frequency domain, a NB-IoT carrier uses one LTE Physical Resource Block (PRB), each of the OFDM symbols consists of 12 subcarriers occupying this way the bandwidth of 180 kHz, hence each subcarrier has 15kHz of spacing [**22, 23**].

In the uplink, NB-IoT supports both single-tone and multitone transmissions. Multitone transmission uses the same SC-FDMA scheme with a 15 kHz subcarrier spacing and a total bandwidth of 180 kHz. In single-tone transmission either 3.75 kHz or 15 kHz are supported. The 15 kHz mode has similar numerology as in LTE. For the 3.75kHz subcarrier spacing, there are, again 7 OFDM symbols within a slot but the symbol duration for 3.75 kHz subcarrier spacing has four times the duration compared to 15 kHz, which results in a slot length of 2 ms. Like the downlink, an uplink NB-IoT carrier uses a total system bandwidth of 180 kHz [**22, 23**].

Advantages:

- Low latency;
- As it uses a mobile wireless network it offers better scalability, quality of service and security compare to unlicensed LPWA networks (LoRa and SigFox).

Disadvantages:

- It offers a lower data rate compare to LTE Cat-M1;
- Deployability could be problematic (deployment in sideband or in deprecated GSM spectrum).

2.3.6. LoRa

Long Range (LoRa) is a low-power wide-area network (LPWAN) technology. It is based on the chirp spread spectrum (CSS) modulation.

LoRaWAN is a communication protocol that uses the LoRa physical layer in order to provide low power long-range communications. LoRaWAN defines the communication protocol and system architecture for the network while the LoRa physical layer enables the long-range communication link [**24, 25**].

LoRaWAN network architecture is deployed in a star topology in which gateways relay messages between end-devices and a central network server. The gateways are connected to the network server via standard IP connections and act as a transparent bridge. A Star architecture is better for preserving battery lifetime than mesh topology. LoRaWAN uses a star topology in order to also maintain the low power long communication viable, reducing complexity and increasing network capacity [**26**].

Advantages:

- Low power consumption;

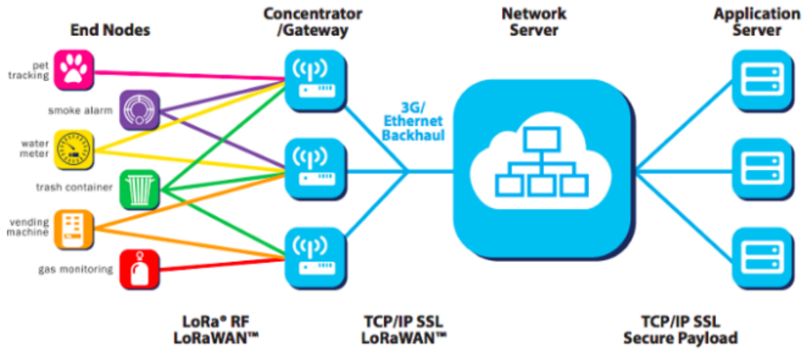

FIGURE 2.6. LoRaWAN Topology

- Long communication range.

Disadvantages:

- Low data rate.

2.3.7. SigFox

SigFox is a low power technology for wireless communication. It allows the transportation of a small amount of data ranging up to 50 kilometers.

SigFox supports unidirectional and bidirectional communication. In Europe, the unlicensed sub-GHz ISM band used for uplink is 868 MHz to 868.40 MHz and for the downlink is 869.40 MHz to 869.95 MHz. SigFox uses Ultra-Narrow Band (UNB) radio transmission for both uplink and downlink. The maximum uplink and downlink transmit power is 25 mW and 500 mW, respectively [**27**].

SigFox employs the Differential Binary Phase-Shift Keying (DBPSK) for uplink, whereas the modulation used for the downlink is the Gaussian Frequency-Shift Keying (GFSK). The uplink physical layer has a maximum data rate of 100 bps, in the case of the downlink physical layer has a maximum data rate of 600 bps [**23, 27**].

In bidirectional communications, the downlink occurs only following the uplink transmission. The number of messages over the uplink is 140 messages per day and the maximum payload length for each of these messages is 12 bytes. However, the number of messages over the downlink is limited to 4 messages per day, with the maximum length for each downlink message of 8 bytes [**23, 27**].

SigFox support star network topology.

Advantages:

- It supports a wide coverage area in the areas where it is located;

- It works well for simple devices that transmit infrequently because it sends very small amounts of data very slowly.

Disadvantages:

- Communication is better headed up from the endpoint to the base station (uplink). It has bidirectional functionality, but its capacity from the base station back to the endpoint (downlink) is constrained.

2.4. Data Analysis

As mention before, the world of IoT is growing more and more, thus with more things connected to the internet, this will lead to an increase in the generated data. In this situation, it is impossible to analyze all the data manually, therefore it is necessary to use technologies with the ability to interpret this data. This is how Machine Learning (ML) arises.

ML is the field focused on building applications that learn from data and improve their accuracy over time without being programmed to do so. In Machine Learning algorithms are 'trained' to inspect data and search for patterns in order to achieve better decisions, Machine Learning combines data with statistical tools to predict an output. The purpose is that the systems learn automatically, with as minimum human intervention as possible [**28, 29**].

ML has various learning techniques that can be used, which the most common are supervised learning and unsupervised learning.

For supervised learning is provided a pair of input-output training data and, this technique will analyse the data and the relationship between them and in the end, it will be possible to predict a function from the input with the best estimation of output [**30**]. A supervised learning algorithm learns from labeled training data, helps you to predict outcomes for unforeseen data.

Supervised learning can be divided into regression and classification [**30**]. In regression, it is predicted a dependent variable Y based on the input independent variable X through statistical processes that estimate the relationship between the variables.

$$Y = f(x) + \epsilon \tag{2.1}$$

In Equation 2.1 is shown the mathematical notation for linear regression where Y is the output, x indicates the input and f is a function that represents the relation between x and Y, ϵ represents a possible random error.

Classification is a predictive model that has the task of approximating a mapping function (f) from input variables (X) to identify discrete output variables (Y). Classification will categorize a set of data into different classes.

$$y(f : x \to y) \tag{2.2}$$

Equation 2.2 shows the function for this method, f represents the mapping function, x input value and y output value.

There are several algorithms in the supervised learning area, but there are some that stand out, the ones that are most used and present the best results include Decision Tree, Random Forest, Neural Network, Support Vector Machine and Linear Regression.

Decision Trees (DT) is a tree-based algorithm that can be used for classification and regression. This method is established on a set of "if-then" rules to enhance readability. The data is split using a hierarchical partition, this split is done iteratively. DT contains two types of nodes: decision nodes, the choice between the existing alternatives and leaf nodes, which are the final outcome. Decision Trees are simple to understand and to interpreted and also reduces ambiguity in decision-making [**30, 31**].

Random Forest (RF) is a tree-based method for classification and regression. In this algorithm, each tree is given a sample. In this algorithm each tree is built from a portion of the dataset, then each tree will give a classification and finally depending on the results of the combination of all trees the algorithm evaluates which is the best option [**32**].

Neural Network (NN) is an algorithm composed of many neurons connected to each other, similar to a human nervous system. This method usually consists of three layers, the input layer, then one or more hidden layers and finally the output layers. Each neuron analyses a part of the input received and sends the information to the neuron in the next layer. The process ends when a final output is found. NN can be used to solve nonlinear and complex problems [**33**]. For the implementation of this technology, it was used Multilayer Perceptron (MLP). MLP is a supervised learning algorithm that learns a function $f(.) : R^m \to R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output [**34**].

Support Vector Machine (SVM) is a set of supervised learning methods that are mostly used for classification and regression. SVMs are effective in high dimensional spaces and use training points in the decision function, so it is also memory efficient. The goal of SVM is to find an optimal hyperplane to categorize the data by classifying the data creating n dimensions between two classes, using the distance between the neighboring points and differentiating between the classes with a minimum error margin [**30, 35**].

Linear Regression (LR) is a supervised machine learning algorithm where the relationship between the input variables (x) and output variable (y) is expressed as an equation of the form $y = a + xb$. The goal of this method is to find out the values of coefficients a and b. In this equation, a is the intercept and b is the slope of the line. These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line [**36**].

2.5. Computing Techniques

There has been a large growth of data that is provided by IoT devices and this is how computing emerges because this growth also increases the need to analyse and store all this data. Computing is the process of using computer technology to complete a task. The two most used computing techniques are Cloud Computing and Edge Computing.

Cloud computing (CC) allows the remote use of resources where data is stored in services that can be accessed from anywhere. As there is no need to have any software installation or additional hardware, all that is needed is a device that has an internet connection, it makes this solution easy and with low investment in servers [**37**].

IoT has grown a lot over the years and this causes devices to generate a lot of amount of data, often these devices use applications that require fast responses or some that produce large amounts of data and Cloud Computing cannot support these applications, and this is how Edge Computing (EC) emerges [**38**]. Edge Computing is a new computing paradigm where IoT data is processed at the edge of the network (cloud edge). Edge Computing aims to move computation from data centers to the edge of the networking to execute tasks closer to the IoT devices and thereby solve some problems such as latency, mobile devices' limited battery life, bandwidth costs, security and privacy [**38**].

2.6. Related Work

In recent years, heterogeneous communication systems deployment has emerged as a new trend to enhance the capacity/coverage and to reduce energy consumption. In our context, wireless networks using different access technologies is what we called a wireless heterogeneous network. For these networks to function it is necessary to use suitable communication in a heterogeneous wireless system considering various parameters such as transmission rate, distance, and bandwidth rather than using the same communication system. Several studies were analysed in order to understand what has been done and what advances have been made in this area and how this project can be innovative.

There are several boards that have some incorporated technologies. One of these boards is a board from pycom, Fipy, where are embed five networks, which are Wi-Fi, Bluetooth, LoRa, SigFox and dual LTE-M (CAT-M1 and NB-IoT), however, this board does not choose autonomously which technology is the best to be used [**39**].

Machine Learning associated with wireless communications also has been grown over time thus several researches can be found in the literature. A great survey of this research spectrum can be found in [**40**], covering several applications from security, interference, link configuration, node locations and several others. Our goal of predicting the link quality based on node location and the type of protocol used, in order to create a heterogeneous communication scheme that switches protocols as a better one is available, falls inside this research. Although, almost none similar approaches were found in the literature, being mainly link or coverage predictions for specific protocols found.

In [**41**], the main objective is to perform a coverage prediction in wireless sensor networks using Machine Learning, to determine an accurate mapping between network features and network performance. It was concluded that the Neural Networks model with three layers was sufficient to achieve high accuracy, and with more than three layers the accuracy did not increase significantly.

The literature also shows some work being done to create a better LoRa link using Machine Learning techniques, with [**42**], using Dynamic Selection, with 96% efficiency, and [**43**] using Neural Networks to improve the energy efficiency of LoRa connection, with a 99.92% accuracy, but only with 200 samples. Some works were also found with the use of machine learning to predict the link quality of BLE mesh networks [**44**].

This proposal aims to solve these problems by creating a heterogeneous communication scheme that evaluates the communication protocols autonomously, and as has been explained, there are already several studies with an approach to Machine Learning in this area of research, however, the theme addressed in this project, as far as we know, has never been developed before, so what we intend to develop here is an innovative project.

CHAPTER 3

System Architecture

This work aims to develop a system that is able to analyse and choose the best communication method autonomously. To implement this system, it was necessary to develop from scratch the software and hardware components.

To develop this project, first, it is necessary to understand the behaviour of the protocols in various circumstances that will be used for the system. After analysing them, with the support of Machine Learning, predictions of signal quality will be made, to then use these Machine Learning models in the final implementation scripts. Machine Learning models will be imported to a script in C++ and this script will be implemented in a communication shield using the ESP32 microcontroller, and a Python script as well, to be implemented in an online platform.

In this chapter, it will be presented the architecture of the system and all the software and hardware that has been developed.

3.1. Methodology

As explained in Section 2.1 one of the main components of IoT is communication. For it to work properly, without reliability and efficiency problems, it is important to use the appropriate protocols and configurations. But with all the evolution in IoT, the devices have become more complex and new protocols are emerging, meaning that not only new solutions will arise, but also new issues might appear. One of the big problems is the lack of ability of the devices to use the adequate communication protocol depending on the scenario they are in.

So, the goal is to create an autonomous communication system that always uses the best protocol available depending on the situation. This system will work for point-to-point communications, communications done directly between devices in range via wireless protocols, and also for cloud communications, communications between devices and cloud servers.

To achieve this goal, first, it is important to understand the behaviour of the protocols in various scenarios that will be used for the system. For that, it is essential to study how different protocols perform under several configurations and scenarios, allowing also the collection of data and knowledge to train the machine learning models.

Then, using the best communication modules tested, a shield will be built in order to attach it to the IoT smart node, in order to allow the node to have multiple choices on how to send or receive messages. This shield will work on a plug&play style, with removable or interchangeable communication modules, in order to create a more adaptable solution.

With the data collected from the communication test and with more data collected online a set of machine learning algorithms will be trained in order to assess which is the best solution for predicting the link quality for wireless communications.

Finally, with all the hardware and software modules developed, the system will be implemented in both cloud computing, taking advantage of a cloud platform and API, that will make available the data analysis models that will help to configure the system, and edge computing, with the models being ported directly into the IoT smart node.

The following sections will describe each of these steps in more detail, showing how they will be achieved and how they are connected to create an autonomous communication system for IoT devices.

3.2. Communication Study

A study and several tests were carried out in order to better understand the behaviour of the most common protocols, analyse in which situations they are best and in which scenarios they present the most limitations. With these tests and with the help of machine learning it will be possible to predict which protocols have a better performance depending on the situation.

The following sections will explain in detail the studies and tests that have been done and how they have been done.

3.2.1. Point-to-Point Communication

A point-to-point communication consists of a direct connection between two parties, which can be two communication endpoints or nodes and can only communicate with each other.

For this system, a point-to-point test was carried out with the objective of studying various protocols, understanding their behaviour and concluding which one presents a better performance. The studied protocols were ESP-Now, BLE, RF and LoRa.

To develop this research, a script was created using the Arduino IDE and C++ and implemented directly on two ESP32 boards, that simulate two distinct nodes. These were used to send a message between them using different communication protocols, under several scenarios and specifications, in order to collect the RSSI.

Several specifications were tested, such as the variation of the number of obstacles between the boards, distance, indoor or outdoor. Each of these specifications was tested for the different wireless communication protocols under study and for each one, it was also tested multiple transmission power values. The combinations that were used between the communication protocols and the transmission power (TxPower) can be seen in Table 3.1 , also the corresponding sensibility for each protocol.

TABLE 3.1. Point-to-Point Protocols Characteristics

Protocol	**Transmission Power (dBm)**	**Energy Consumption (mA)**			**Range(m)**		**Sensibility (dBm)**
		Transmit	**IDLE**	**Sleep**	**Indoor/ Urban**	**LoS/ Rural**	
ESP-Now	1	90	60	-	100	500	-98
BLE	-12, -6, 3, 9	7, 8, 9, 10	10	0.07	100	250	-94
RF	14, 16, 18, 20	50, 80, 120, 150	16	0.001	200	500	-120
LoRa	5, 7, 10 15, 20, 23	20, 35, 65 80, 100, 120	12	0.001	500	2000	-135

For the outdoor scenario, the measurements were taken on a street with no obstacle interfering with the line of sight, as shown in Figure 3.1. The following distances between the two nodes were tested: 1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 120, 140, 160, 180, 200 meters. For the indoor scenario, a 118 m^2 house was used, as shown in Figure 3.2. The transmission node, the red dots, was always in the same place and several other points, the green dots were tested around it. With points in the same room, i.e. without any obstacles between them, and then, with other points in different divisions, which include walls, doors and furniture between the two nodes as obstacles.

3.2.1.1. *Outdoor.* Figure 3.3 shows the performance of the various technologies used in this experiment for the outdoor scenario.

As mentioned before, these measurements were made with no obstacles in the line of sight, and the distance between the two boards is up to 200 meters, however, as expected only LoRa was able to obtain results at that distance since among the four technologies LoRa is the one that theoretically has a greater range. BLE was the protocol with the

FIGURE 3.1. Outdoor Scenario

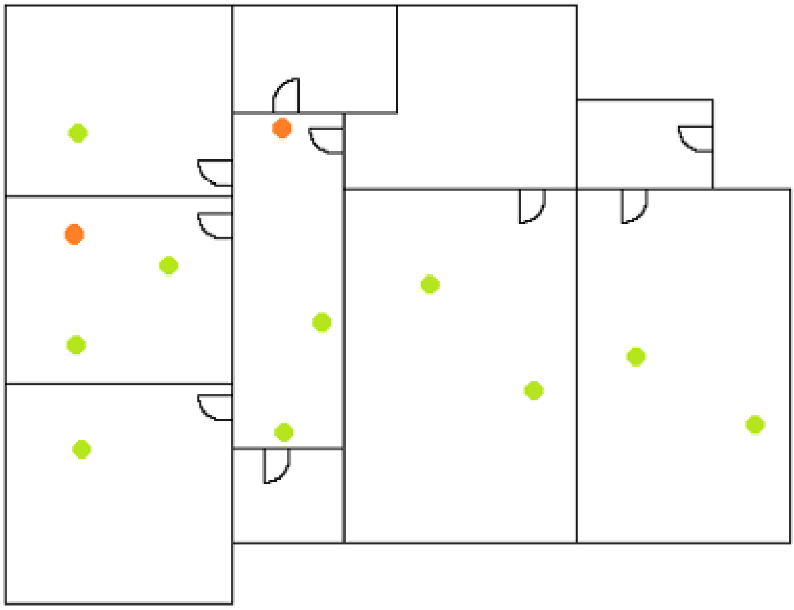

FIGURE 3.2. Indoor Scenario

most limitations to transmit the signal, with a transmission power of -12 dBm where the difficulties were, being only able to receive messages up to a distance of 50 meters and with an average RSSI value close to the sensibility threshold, around -87.7 dBm.

Taking into consideration the distance at which the protocol can transmit and the RSSI value, the technologies that present a better result were the ESP-Now with a value around -75.3 dBm but can only transmit to a distance of about 180 meters, on the other

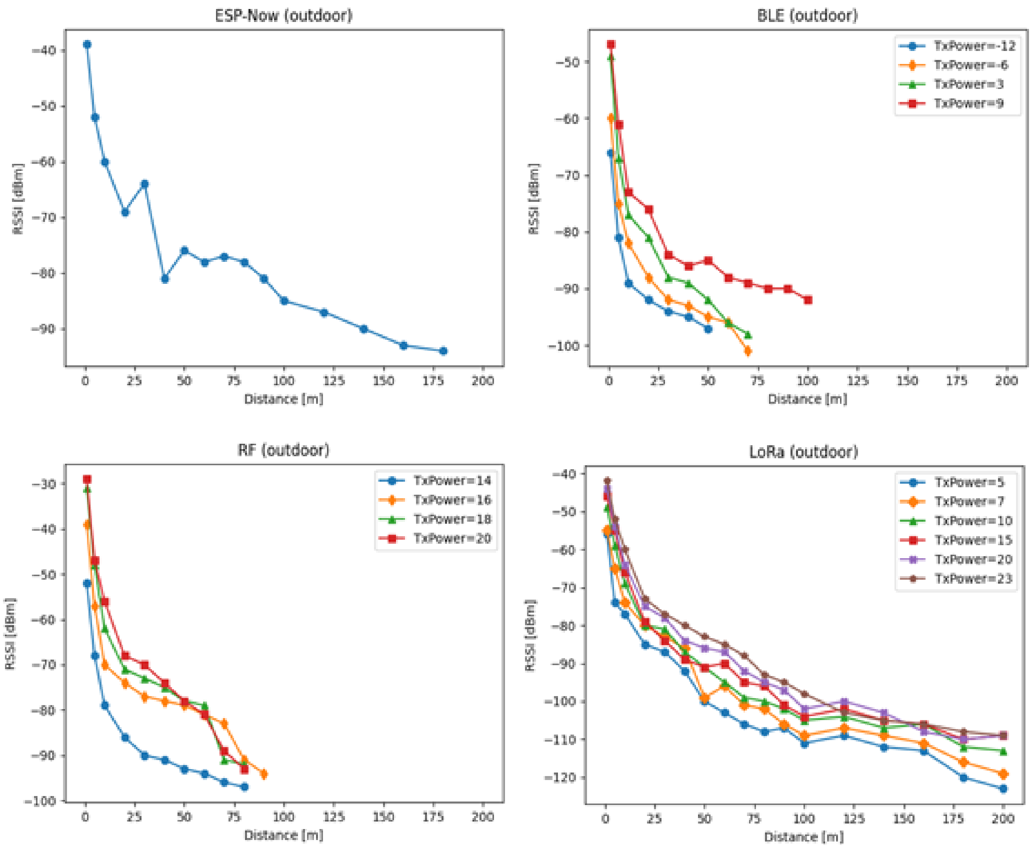

FIGURE 3.3. Outdoor Scenario Results

hand, LoRa with a transmission power of 23 dBm can transmit to at least 200 meters but with an average RSSI value of -85.7 dBm.

Therefore, although LoRa uses more energy, about 33.3% more than the ESP-Now, it will end up being the best protocol to use for this type of scenario where there are large distances, because although ESP-Now uses less energy and has better RSSI, it is not possible to obtain data when the distance becomes considerable.

3.2.1.2. *Indoor.* For the indoor scenario, Figure 3.4 shows the performance of the various technologies used in this experiment.

For these tests, as the distance increased, so did the number of obstacles, so in all tests, there was a considerable decrease in the RSSI values.

It is possible to observe that almost all protocols were able to transmit at all distances, except for the BLE, as it happened in the outdoor scenario, was the one that presented the most limitations, which when its transmission power was -12 dBm, was not able to receive any signal when the distance was more than 7 meters.

Two measurements were made with the boards 2 meters apart, in the first one there were no obstacles in the line of sight, and in the second one there was an obstacle, thus in

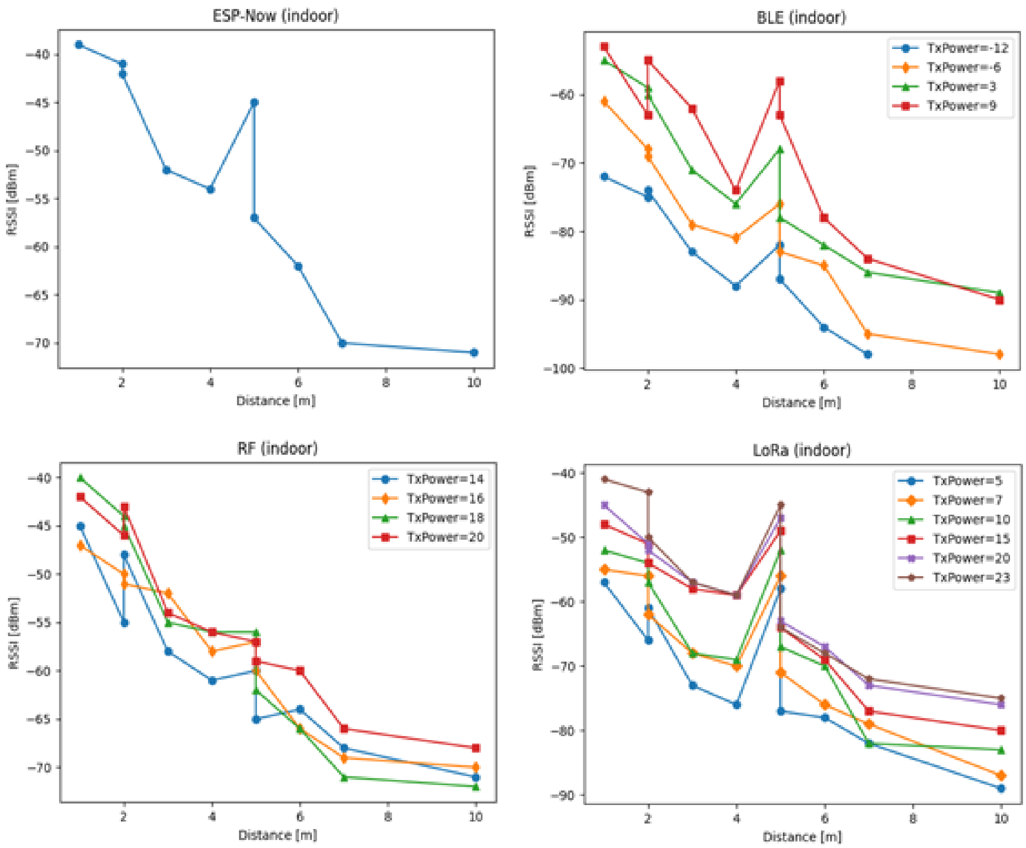

FIGURE 3.4. Indoor Scenario Results

most cases, as expected, the first measurement has a slightly lower RSSI than the second one. The same behaviour happens in the peak that can be seen in all graphs at the 5 meters, in the first there were no obstacles and in the second there were two, hence the considerable decrease in the quality of the signal.

The technology that performs best in this scenario is ESP-Now. Although RF and LoRa can also transmit at these distances, they can consume up to 66.7% more energy than the ESP-Now, having a lower RSSI, and are better suited for longer distances, so ESP-Now turns out to be the best solution for this type of scenario. ESP-Now is the one that on average gets the better RSSI, about -53.3 dBm, and the one that presents the worst behaviour is, as already mentioned, BLE when the transmit power is -12 dBm because besides not being able to receive any message from 7 meters is the one that presents the lowest RSSI values, with an average of -83.7 dBm.

3.2.2. Cloud Communication

There has been a big growth in the use of outdoor devices, and what used to be the most common, small connections such as in homes, has now become connections across an entire city, or even different cities connected together. With this, the devices have the need to exchange messages to cloud servers for analysis and to be able to communicate with other networks as well. Thus, new communication protocols with a greater range than, for example, ESP-Now or BLE, have emerged. In the system discussed in this thesis that was developed for cloud communication, the protocols used were, 2G, 3G, 4G, Wi-Fi, LoRaWAN and SigFox.

For this scenario, it was not necessary to perform tests to understand how the signal is affected by obstacles or the conditions in which the device is located as was done in the point-to-point scenario.

Cloud communications will depend on antennas that are distributed throughout the city or region, and these antennas are already developed to fix the problems and limitations that were studied in the previous scenario. For these reasons, for the algorithm to decide the best protocol, it is more important to know the protocols that are available at the location of the device rather than the local conditions.

3.3. Communication Shield

Regarding the hardware for this implementation, it consists of a communication shield where the goal is to detect which radios modules are connected. The shield is composed of several communication modules, that can be swapped or used simultaneous, including the RFM95W transceiver for LoRa and LoRaWAN, the RFM69HCW transceiver for RF, and the SIM7000E for cellular communications; and an analog multiplexer, the 74HC4051, to create an identification mechanism to understand which modules are attached. The shield will be assembled in a perforated board, to be able to be attached to the IoT node.

The IoT node is composed of an ESP32-DevkitC-v4 development board, that includes the ESP32 microcontroller.

ESP32 [**45**], is a low-cost, low-power system on a chip series of microcontrollers with integrated Wi-fi and Bluetooth. ESP32 is an upgrade from his predecessor, ESP8266. It offers a better solution to be implemented in more complex projects. ESP32 has several power modes that can operate; active mode; modem mode – CPU is operational but Wi-Fi and Bluetooth are powered off) and deep sleep mode which is the mode that allows to lower the power consumption of the board [**46**].

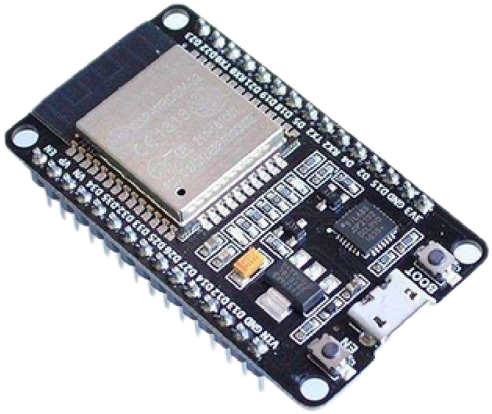

FIGURE 3.5. ESP32 board

3.4. Data Analysis

For the user to be able to access and visualize the data in real time several scripts need to be implemented. The main goal of these scripts is to use the received data and, with the support of Machine Learning algorithms, return the desired results, such as the best communication protocol or its configuration.

The following sections will explain how these scripts were developed, implemented and how Machine Learning was used to obtain the results.

3.4.1. Decision Scripts

As explained, the entire goal of this project is to have a decision process capable of understanding the best communication protocol to use under certain conditions and based on the available protocols, for both point-to-point and cloud communications.

To achieve that, for the Point-to-Point scenario, a script in Python, named script_p2p.py, will allow the user to input the various parameters, the device position, the scenario (indoor or outdoor), the number of obstacles and the protocols, that can be manually inputted or given by the IoT node detection mechanism. The script outputs the protocol and transmission power that is associated with the best configuration.

For the Cloud scenario, a script named script_cloud.py, in the same fashion as the point-to-point will allow the user to provide the parameters, the coordinates (Longitude and Latitude) of the device and the protocols. The scripts outputs the best protocol associated with the given conditions.

In Figure 3.6, can be seen the methodology used for the creation of these scripts.

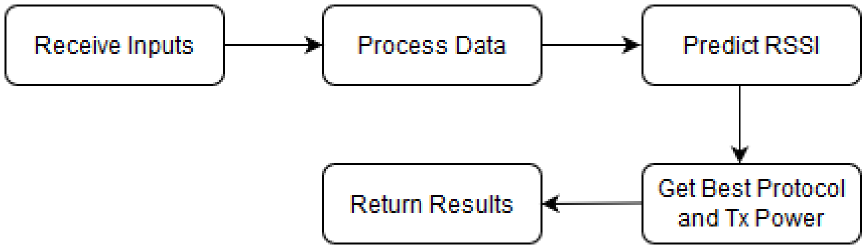

FIGURE 3.6. Scripts Creation Methodology

3.4.2. Machine Learning

As mentioned before the goal of this system is to understand which is the best protocol to use depending on the scenario. To make the system functional and efficient it is necessary to make a prediction of the RSSI based on the provided values. To achieve this machine learning will be used.

The scripts that will be developed, as mentioned in the previous section, will implement a machine learning algorithm, that when receiving the inputs, return the best protocol.

The methodology used for this machine learning approach is explained in Figure 3.7. The first step, after data collection, for the system to be able to make this prediction is the creation of a dataset, then it must be trained with various Machine Learning algorithms when we know which is the best algorithm, this same algorithm will be implemented into the script that was previously created and then using the predict() function it will be possible to obtain the RSSI predictions.

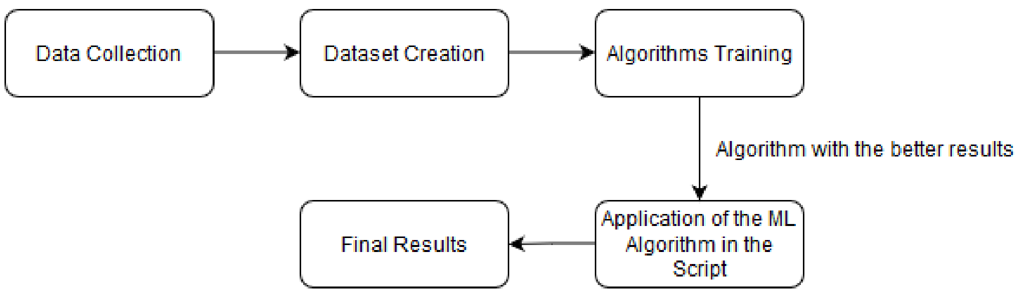

FIGURE 3.7. Machine Learning Approach Methodology

The creation of the dataset, the study and the choice of the best model to use will be explained in detail in Chapter 4.

3.5. Support Platforms

Regarding the software part of the system, for the user to be able to access, visualize and monitor the data, an API and an online platform will be used.

These platforms are how the user can interact with the previously described Python scripts, explained in Section 3.4.1, that are implemented in the backend of these support platforms. The next sections explain how was the procedure to create these platforms.

3.5.1. Applications Programming Interface

An Applications Programming Interface (API) is a set of definitions and protocols for building and integrating application software. APIs allow an application to communicate with others without having to know how they are implemented.

For building the API that allows the user to connect to our endpoint and retrieve the information about the best communication protocol and configuration for their specifications and scenario, Flask, a micro framework to create web applications written in Python, was used.

Two endpoints were created, one for each scenario, each receiving the necessary inputs, as described earlier. These endpoints work via request messages sent through HTTP packages. HTTP implements some methods, being GET, POST, PUT and DELETE the most commons, but for these implementations, only the GET method was used.

The developed endpoints are:

- "http://127.0.0.1:5000/api/point2point?X=10&Y=4&scenario=2&obstacles=3&protocols=1,2,4" – for the point-to-point scenario;
- "http://127.0.0.1:5000/api/cloud?latitude=51.0121232&longitude=3.7136986&protocols=1,2,3" – for the cloud scenario;

The API Flask server waits for incoming requests and when a request is received, with the format that was defined with the required parameters, these parameters are extracted from the URL created by Flask and the decision scripts, implemented alongside the Flask server, will analyse and decide, returning a JSON array with all the desired data.

3.5.2. Online Platform

Although the API is capable of providing the needed information and results, it was developed to be used by the IoT node, to make requests directly using HTTP.

In order to be able to allow the user to have a more interactive experience and to visualize and analyse the data, a web application was developed from scratch using Flask, for the web app server, and HTML and CSS for the design.

As in the API, the web application implements the decision scripts in the backend, being the parameters given by a set of web pages with forms for each of the scenarios. As such, the web application has the following pages:

- '/' – On the main page it is presented two options. It is possible to choose which scenario we want: Point to Point and Cloud;
- '/point_2_point' – If the 'Point to Point' button is selected, Flask will receive a request for this URL and will display a page with several fields to fill in, then when the 'Calculate' button is pressed it will return the results (best RSSI, best protocol and best TxPower);
- '/cloud' – If, in turn, the 'Cloud' button is selected, the request will be made to this URL and the page with the fields referring to this scenario will appear and after filling the fields, as in the previous scenario, the results will appear (best RSSI and best protocol).

Once again, as the decision scripts were implemented alongside the Flask server, they were responsible for analyse and decide the best protocol and its configuration based on the parameters given by the user.

CHAPTER 4

Machine Learning Training

The main goal of this project is to create a heterogeneous communication scheme capable of using the most common communication protocols in IoT to be connected to an existing Smart Node, that can be applied in any environment, and predict which is the best technology to use according to the scenario. The autonomous communication scheme is composed by two separate systems, one for point-to-point communication and another for cloud communication, and each of them will function based on the prediction for the best communication protocol. These decisions will be achieved with the use of Machine Learning techniques.

As described before, Machine Learning is a technology with the ability to predict future data based on previously known data that is trained in order to create a model capable of extracting knowledge and future prediction from that data.

To achieve the goal of predicting the best communication protocols based on the implementation scenario, a set of Machine Learning algorithms will be tested for the prediction of communication behaviour in several different environments and specifications such as indoor/outdoor location, line of sight, number of obstacles, quality of the signal, location and distance to the receiver. With the obtained results it will be possible to analyse which of the tested algorithms are the best to use for the creation of the autonomous communication scheme.

To study the most suitable algorithm that can be used to do regressions, it was considered the following Machine Learning algorithms: Random Forest (RF), Neural Networks (NN), Decision Trees (DT) and Linear Regression (LR).

In this chapter, it will be presented the research and development related to Machine Learning, from the methodology that was used for the study, the datasets that were used for the different scenarios and how they have been built, the training process and the obtained results.

4.1. Training Methodology

This methodology uses a regression model for the computation analysis, and as described before, for this model to work it needs to be trained with previously known data and configured to achieve the best accuracy and lowest error possible. So, in order to train the regression model, the following steps were done, using Python, the scikit-learn libraries [**47**] and the Anaconda environment.

(1) For each algorithm, a model was trained using the corresponding dataset and the default configuration parameters. This allowed for a quick comparison of the performance, in terms of both accuracy and margin of error, of each model and understanding which are more likely to guarantee the best results and which need to be improved to achieve them. The scikit-learn, an open-source Machine Learning library developed for Python implementation [**47**], was the selected framework for the development of the machine learning models;
(2) The obtained model for each algorithm is submitted into a hyper parametrization tuning, that compares the model performance using different model configuration parameters, to understand which is the configuration that obtains the best performance, facing the dataset and the goal. For this, a method provided by scikit-learn called RandomizedSearchCV was used, which performs the fit and training of the algorithm under study, calculating which parameters are best suited to it [**48**];
(3) To guarantee that the model is stable, after finding the best configuration and training the model, a Stratified KFold cross-validation is performed, in order to guarantee that the model is not under or over fitted. Using five folds, it is possible to use a different set of training and validation data on each fold, allowing for the model to check on every single datapoint. This way, it is possible to really understand the model performance, as each of the folds will produce a result, that is averaged at the end, allowing for a reduced error margin and variation, as more data is used to fit the model.

This methodology will be used for both of the regression models that will create the autonomous communication scheme, for point-to-point and cloud communication.

4.2. Dataset

To implement a regression model, that is included in the supervised learning category, a set of previously known data is needed. For a Machine Learning model to work and to have a better understanding of which model works better in as many situations as possible the used dataset must include the output for all the possible sets of parameters, as it is with these data that the model will learn and predict future data, and without a specific output in the dataset, it is impossible for the model to predict that output.

Since two distinct models need to be created, and each one uses a different type of features and parameters, two datasets were used. Each of them is described in detail in the following sections, including how it was gathered, the included features and the goal to achieve with that data.

4.2.1. Point-to-Point

Point-to-Point communication aims to create the best link between two or more devices that can be distant and have obstacles between them. To surpass these conditions the communication modules can be configured to have different settings, such as transmission power, baud rate, among others, that when tweaked can improve the communication distance and resilience to obstacles. The creation of this dataset aims to evaluate how each of these configurations performs under certain conditions and protocols, in order to understand which Machine Learning regression model has the best accuracy in predicting signal quality.

In this scenario, the gathered information for the point-to-point comparison test described in Section 3.2.1 was used, as it includes information about the transmission of data packets using point-to-point protocols.

The final dataset is composed of 346 entries with the following parameters:

- X – Transmitter position in meters;
- Y – Receiver position in meters;
- Scenario – Scenario that is being used in the data transmission: 1 – outdoor, 2 – indoor;
- Distance – Distance between the two boards in meters;
- Obstacles – Number of obstacles between the two boards;
- Protocol – Protocol used in the data transmission: 1 – ESP-Now, 2 – BLE, 3 – RF, 4 – LoRa;
- TxPower – Transmission Power;
- RSSI – Link quality of the data transmission.

4.2.2. Cloud

Cloud communications aim to create a link between a device and the cloud servers, using a gateway, Wi-Fi router or cellular tower, that might be distant from the device.

As in the point-to-point communication tweaking the configuration of the communication module might improve the communication link and knowing how each configuration performs will help in the decision process. The main objective of this scenario is to study the regression models, in order to create a model capable of receiving as input the location of the node (Latitude and Longitude) and the communication protocol to use and output the predicted quality link.

The achieve that goal the used dataset was composed of crowd-sourced data collected by the NetBravo project, [**49**]," a European Commission crowd-sourcing project designed to gather and share radio spectrum data about mobile telephony coverage, Wi-Fi channel occupancy, broadband and net neutrality connection tests. Anyone with a recent smartphone can download the netBravo app which will automatically record the characteristics of the signal they're getting on their phone – Wi-Fi, 4G, 3G, 2G or nothing – and test the latency, upload and download performance of their Internet connection with additional net neutrality tests they can select.", combined with LoRaWAN and SigFox data from a fingerprint localization dataset for large outdoor environments [**50**], that contains information such as "the receiving time of the message, base station IDs' of all receiving base stations and the Received Signal Strength Indicator (RSSI) per base station".

This allows us to have a dataset of multiple cloud wireless protocols such as 2G, 3G, 4G, Wi-Fi, LoRaWAN and SigFox. The NetBravo dataset contains points from across the entire European Union countries, whereas the other dataset only contains data from Antwerp, Belgium. As such, to have a more reliable result, only the NetBravo entries from the Antwerp region were considered in the final dataset. With that in mind, the final dataset is composed of 849517 entries, with the following parameters:

- Latitude – Latitude of the node;
- Longitude – Longitude of the node;
- Protocol – Protocol used in the data transmission: 1 – SigFox, 2 – LoRaWAN, 3 – 2G, 4 – 3G, 5 – 4G, 6 – Wi-Fi (2.4GHz), 7 – Wi-Fi (5GHz);
- RSSI – Link quality of the data transmission.

4.3. Results

The presented training methodology was followed in order to obtain the best model possible to predict the link quality of data transmissions, based on node position and link configurations, for both point-to-point and cloud communication.

To train, validate and test the model, the presented datasets were used, being divided into three groups: 70% for training, 20% for validation and 10% for testing. To evaluate the models performance, and since regression is used, the Mean Absolute Error (MAE) metric will be used, as it is the most common metric for regression. It measures the average absolute error between the real data and the estimated value, using Equation 4.1 [**51**], where P_{rx} is the real value, $\widehat{P}_{rx}$ is the estimated value, and N is the number of samples.

$$MAE\ [\text{dBm}] = \frac{1}{N}\sum_{i=1}^{N}|P_{rx_i} - \widehat{P}_{rx_i}| \tag{4.1}$$

The estimated data nearly matched the real data when MAE is near 0.

4.3.1. Point-to-Point

For the point-to-point communication scenario, which intends to predict the link based on node location, the distance between nodes, obstacles, used protocols and its configuration based on transmission power, the regression models can be seen in Figure 4.1.

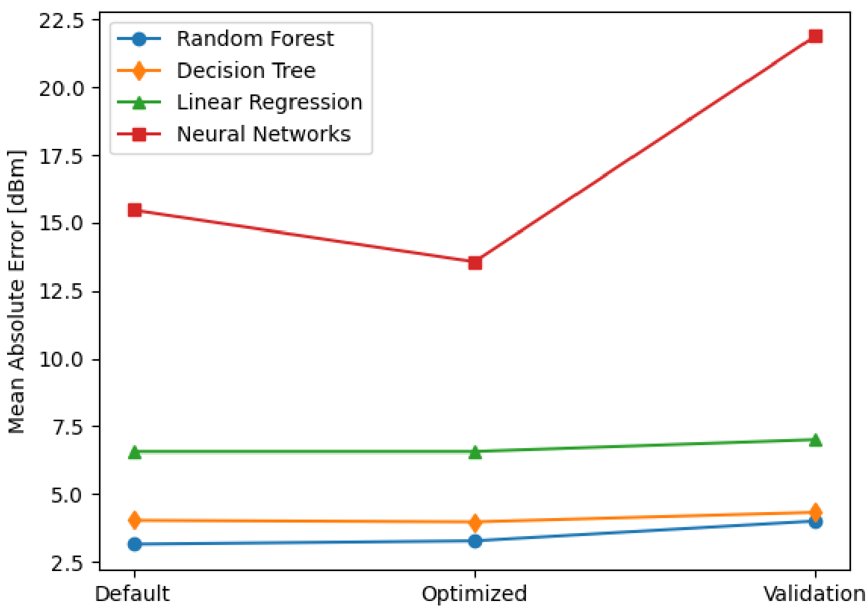

FIGURE 4.1. Point-to-Point Regression Models Results

As is possible to see the values have a noticeable variation depending on the type of the test. Analysing the results, it can be seen that the hyper parametrization values are the ones that have the lowest MAE values, except when the Random Forest algorithm was used where the hyper-parametrization has the value slightly higher than the default value, and the cross-validation has the highest values, and the default values are in the middle.

Comparing the results, it can be concluded that the best algorithm is the Random Forest with a margin of error of 4.018 dBm after the cross-validation phase.

Random Forest when compared with the Decision Tree, the second best model, it achieves a lower MAE by 0.706 dBm and 0.323 dBm, in the optimized and cross-validation phases, respectively. Neural Networks presented the worst results among all models, obtaining very high values.

In terms of network performance, according to [**51**] a good propagation is reached when the model has a margin of error lower than 9 dBm, so Neural Network is, definitely, not appropriate for these tests.

Random Forest achieved the best results when using the following parameters:

- n_estimators: 200
- min_samples_split: 5
- min_samples_leaf: 2
- max_features: ‘auto’
- max_depth: 50
- criterion: ‘mse’
- boostrap: True

4.3.2. Cloud

For the cloud communication scenario, with the goal of predicting the link quality of messages exchanges based on the node location and the available protocol, the results obtained using the regression models, for each step of the presented methodology, are displayed in Figure 4.2.

As in the point-to-point scenario, the values have a noticeable variation depending on the type of test. Analysing the results, it can be seen that the hyper parametrization values are the ones that always have the lowest MAE values, then the cross-validation and finally the default values with the highest values, except in the scenario where the Neural Network algorithm was used, where the cross-validation has the value slightly higher than the default value. This follows the expected behaviour due to the presented methodology since, after the first test with the default values, these values will be tuned to improve, being predictable that in the optimized scenario better results will be obtained. Then, in

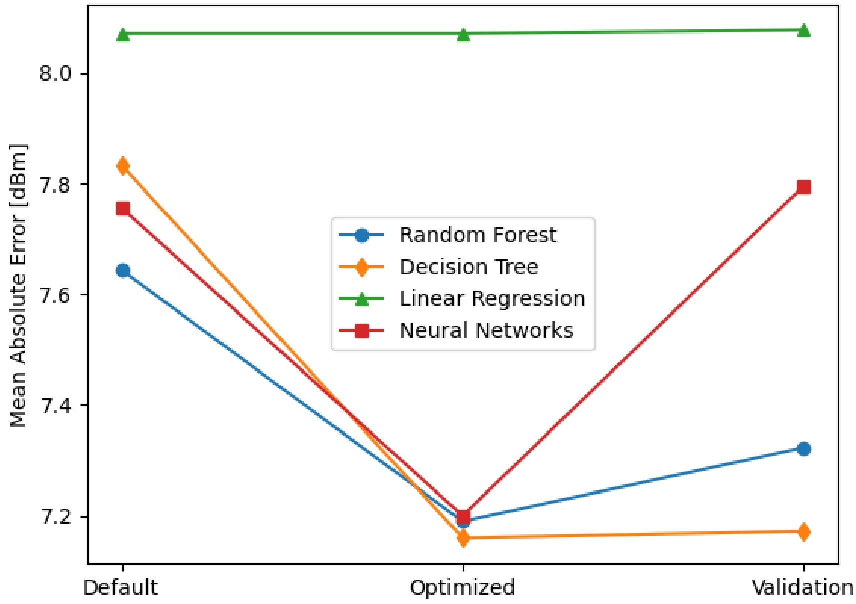

FIGURE 4.2. Cloud Regression Models Results

the cross-validation multi combinations of the dataset are tested and the MAE for each fold is averaged, thus it is expected that the MAE increases.

Comparing the results obtained across the models, it is possible to conclude that the best algorithm is the Decision Tree, with a margin of error of 7.172 dBm, after the cross-validation phase.

Decision Tree when compared with Random Forest, which is the second best model, it achieves a lower MAE by 0.030 dBm and 0.151 dBm, in the optimized and cross-validation phases, respectively. As described, Neural Networks have a cross-validation MAE higher than the default and optimized stage, which means that the model does not work well with unseen data. Regarding Linear Regression, as expected, presented the worst result among all models. As said before, a good propagation is reached if the model had a margin of error lower than 9 dBm, so Random Forest is able to create a good prediction model for link quality in a communication network.

The best results are achieved with a Decision Tree model with the following parameters:

– splitter: 'random'

– min_samples_split: 10

– min_samples_leaf: 4

  - max_features: ‘auto’
  - max_depth: 20
  - criterion: ‘friedman_mse’

CHAPTER 5

System Implementation

After explaining all the system architecture, how the essential data was collected for the development of the project, all the studies and tests done, the scripts needed, the application of Machine Learning and also the hardware in Chapter 3, this chapter presents how the implementation of the system and the development of the smart node will be done.

As already mentioned, the goal is to develop a smart node that can operate without human intervention that has the ability to self-configure the communication protocol that is used for exchanging messages between nodes and with the cloud, based on its locations, environmental conditions and attached modules. Edge computing algorithms will help the smart node in this task. The algorithms will detect which technologies are connected to the node and will configure it as needed.

This chapter will start with a description of how machine learning models were adapted to be used in an edge computing analysis, how the detection of the modules was done by the communication shield, and how the node can also work using the developed API instead of using edge computing. For each of these situations, the implementation tests made, and their results will be described and analysed in order to understand which of the approaches achieves a better decision time, lower power consumption and better accuracy.

5.1. Machine Learning Models Configuration

For the implementation of this system, the developed and trained models presented in Chapter 4 will be used. For point-to-point communications, the Random Forest model, which predicts the best communication protocol and its configuration based on location, indoor or outdoor environment, distance and obstacles between the nodes, with a margin of error of 4.018 dBm, will be implemented in the node. For the cloud communication, it will be used the Decision Tree model, which predicts the best protocol available based on location, with a margin of error of 7.172 dBm.

As described in Section 2.5 data analysis can be done in several points along with the IoT system, from the cloud to the edge devices. Although the developed system includes a cloud server and an online platform capable of running the decision algorithms, in order to evaluate if a truly self-configuration smart node is capable of being deployed, without the need for additional infrastructures, the system was designed to implement the decision models directly on the nodes.

With the help of edge computing, the developed Machine Learning models were used in the microcontroller of the IoT node, with the goal of analysing the attached modules in real time and adjust the communication system based on its best configuration, according to the model predictions.

As the edge computing scripts will run on the microcontroller, the models needed to be ported into a C file, instead of the Python file resulted from the training methodology presented in Chapter 4. To do this the micromlgen library was used and thus a C file was created that could run on the microcontroller.

However, the microcontroller used in these smart devices have quite limited Flash, for example, the ESP32 used in our smart node has only 2 Mb of available Flash to store additional files, such as edge computing models. When analysing the two ported models, with the best configurations, was possible to assess that both had a size too large for the microcontrollers capacity, the Random Forest model for the point-to-point scenario had a size of 3.15 MB, and the Decision Tree for the cloud size of 2.06 MB. Therefore, some modification was needed to slightly reduce their sizes.

Among the various parameters that can be considered for the Random Forest, the depth and the number of estimators are the ones that can affect the final size of the model the most, since they define the number of trees and the size of each tree. For the Decision Tree case since there is no number of estimators parameter only the depth was changed.

As it was possible to conclude in the training phase, tweaking with the parameters might result in a higher or lower MAE, so it is important to guarantee that not only a smaller file can be achieved in order to run it directly in the microcontroller, but it is also necessary to continue with a model capable of having a good accuracy and reliability. Table 5.1 shows the parameters configuration of the best trained models in Chapter 4, as well as the resulting MAE.

Regarding the Random Forest model, with the configuration parameters mentioned in Section 4.3.1, first the number of estimators was changed to check how it affects the model size. Staring with the model original of 200 and decreasing the number of estimators, in Table 5.2 it can be observed that there were no significant variations in MAE, and that

Table 5.1. Best Trained Models Parameters

Model	**Size (MB)**	**Estimators**	**Depth**	**MAE (dBm)**
Random Forest	3.15	200	50	5.217
Decision Tree	2.06	-	20	8.579

it was originating a smaller file. Continue to decrease the number of estimators until it obtains a file size smaller than 2 MB, it was possible to create a model with a file 42.2% smaller, with the number of estimators decreased by about 42.5%. This affected the model MAE as it was expected, but in the end, it only increased by about 0.28%, not putting at risk the model accuracy and efficiency. Table 5.2 shows all the results obtained.

Table 5.2. Estimators Impact on the Random Forest Model

Feature	**Results**			
Estimators	200	150	120	115
Depth	50	50	50	50
MAE (dBm)	5.217	5.290	5.246	5.232
Size (MB)	3.15	2.37	2.01	1.82

It was also necessary to experiment decrease only the depth parameter and keeping the number of estimators always the same to see if it was possible to get better results than before.

Table 5.3. Depth Impact on the Random Forest Model

Feature	**Results**			
Estimators	200	200	200	200
Depth	50	20	10	5
MAE (dBm)	5.217	5.217	5.219	6.464
Size (MB)	3.15	3.15	3.03	1

As we can see in Table 5.3 decreasing the depth does not change the results that much, the file size only decreases significantly when the depth is reduced by about 80% and even gets a smaller size than when the estimators are decreased, but the MAE increases too considerably, by 28.89%, which is still quite significant. Hence, in the end, it was decided to decrease only the number of estimators and keep the depth unchanged.

For the Decision Tree model, it was necessary to reduce the max_depth parameter by 50% to get a satisfactory result. The file became 93% smaller with a very minor variation of MAE, about 1.43%. Table 5.4 shows the results of the parameter modifications.

Table 5.4. Depth Impact on the Decision Tree Model

Feature	**Results**	
Depth	20	10
MAE (dBm)	8.579	8.702
Size (MB)	2.06	0.144

In Table 5.5 we can see how the models turn out after the adjustments.

Table 5.5. Final Models Parameters

Model	**Size (MB)**	**Estimators**	**Depth**	**MAE (dBm)**
Random Forest	1.82	115	50	5.232
Decision Tree	0.114	-	10	8.702

For the Random Forest model, the various parameters were studied, mainly, decreasing the depth and number of estimators. It was concluded that when changing the depth, although it does not affect the MAE, it neither affects the size. Therefore, after changing various parameters, it was possible to verify that changing the n_estimators parameter would be the best, it was where a greater difference in file size was effective, without a higher increase in the MAE.

Regarding the Decision Tree model, the same studies were done, and unlike the previous model, the biggest change in the file size occurred by decreasing the max_depth parameter without increasing the error margin excessively.

5.2. Communication Shield

For the development of this smart node, it is important to be able to connect the various radios, for this a shield in a plug&play system was developed, where it is possible to have various radios connected at the same time, turn them on and off, and easily replace one radio for another.

In order to implement this system and achieve the objectives of this thesis the shield must allow associating to the smart node radios for LoRa and RF, for the point-to-point, and cellular, LoRaWAN and SigFox for the cloud, since the ESP32 already has built-in Wi-Fi, ESP-Now and BLE, as already mentioned in Section 3.3.

Since the shield uses a plug&play and interchangeable modules, there is a need to know what modules are attached. For that, a multiplexer, in this case, the 74HC4051, is used, being each of its ports associated with a specific module. When a module is attached to the communication shield a 3.3V signal is sent to its specific port, allowing for a later analysis of which modules are attached.

For the point-to-point four ports were used:

1 – ESP-Now; 2 – BLE; 3 – RF; 4 – LoRa.

And for the cloud scenario, it was used seven ports:

1 – SigFox; 2 – LoRaWAN; 3 – 2G; 4 – 3G; 5 – 4G; 6 – Wi-Fi (2.4GHz); 7 – Wi-Fi (5GHz).

In Figure 5.1 it is represented the communication shield.

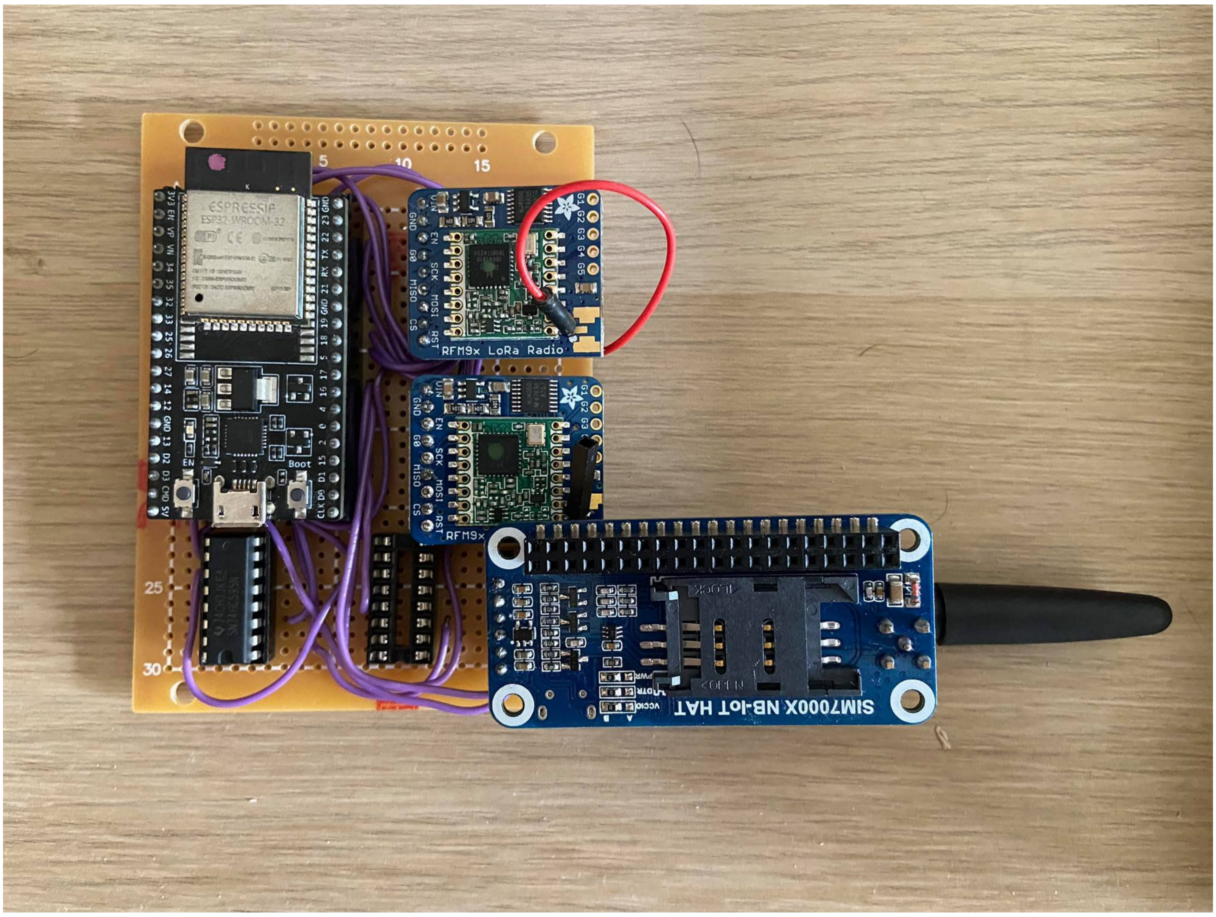

Figure 5.1. Communication Shield

5.3. Module Detection

As already mentioned, the smart node will know which protocols are connected and for this, it will be necessary to develop a detection mechanism, as the algorithms will depend on knowing which protocols are available to be used and as such it is important to know in real time which modules are connected.

To achieve that, a set of scripts was developed and implemented alongside the main smart node script, according to the methodology described in Section 3.3, in order to achieve the goal of the smart node being able to collect and analyse the information about the attached modules and then decide how it will configure itself.

As described the communication shield includes a multiplexer responsible for collecting the information about the connected modules, being that information capable of being retrieved by asking the current value of a certain port. As such, the detection mechanism is controlled using a script, developed in the Arduino IDE with C++, that is running in the IoT node with the objective of collecting data about the modules that are connected to the board. The script verifies which inputs are flagged as HIGH, or have the analog value 4095, meaning that the corresponding communication module is attached. With this, the IoT node is capable of knowing all the modules that are connected.

In Figure 5.2 and Figure 5.3 it is possible to see an example of the result of the script that will be demonstrated in the Arduino IDE console with the radio modules that are attached. In this case, as it possible to observe, all the modules are attached.

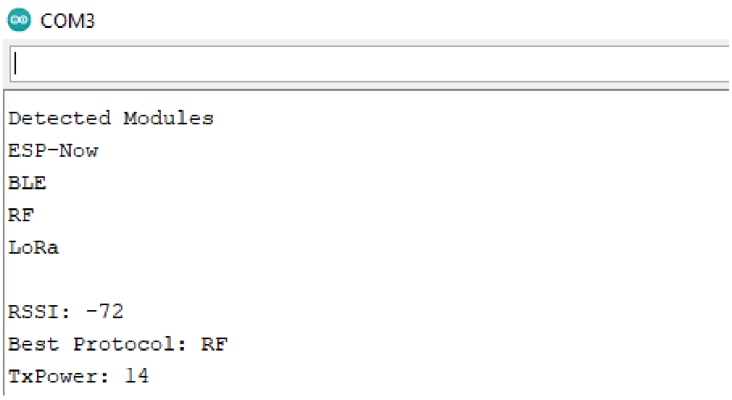

FIGURE 5.2. Attached Modules for the Point-to-Point Scenario

5.4. Self Configuration

After the development of all the required modules to create the self-adjusting node, it is necessary to create a mechanism that brings the various modules together so that the configuration process is automatic when the node is powered on.

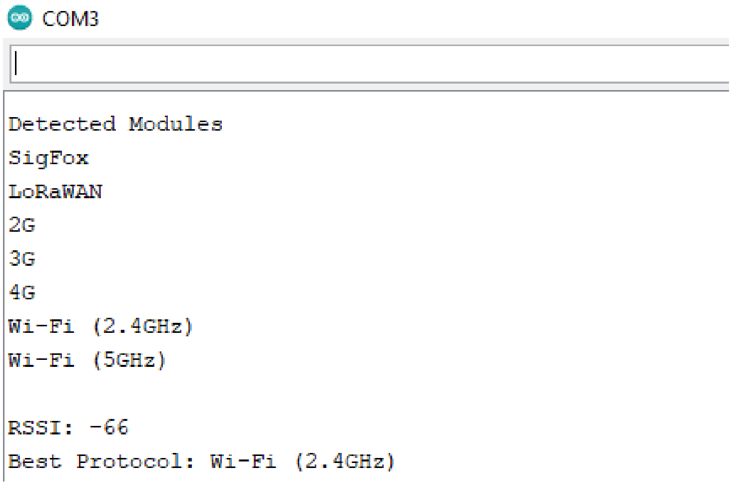

FIGURE 5.3. Attached Modules for the Cloud Scenario

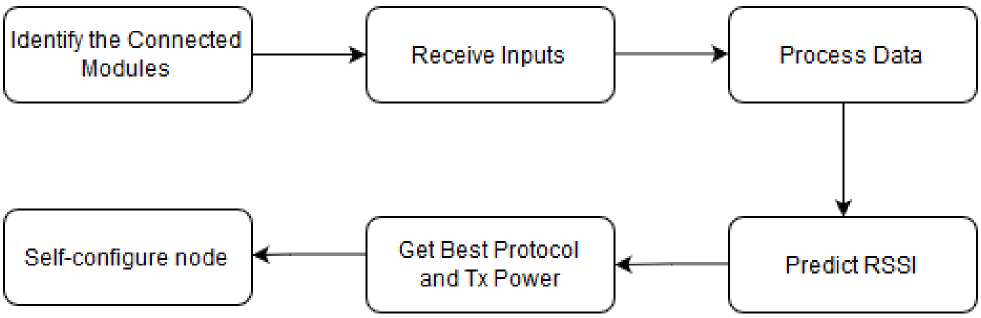

FIGURE 5.4. Self-Configuration Mechanism

In Figure 5.4 we can see a better representation of how the self-configuration mechanism will work when the node is powered on.

First, it will detect which modules are connected, using the module detection mechanism described earlier. Then it will receive the inputs that the user will insert, from the position of the device, the scenario and the number of obstacles, and after processing that information, it will predict the RSSI with the help of the machine learning models. This prediction can be made either using API or edge computing. With the predicted RSSI values, an analysis will be made to be able to understand which module is the best and then automatically configure the smart node with the best technology.

5.4.1. RSSI Prediction

For the prediction of the RSSI for the available protocols and based on the inputs given by the user, both edge computing and cloud computing, via an API, can be used.

In edge computing, the input data is encoded directly into the device, being that the location can also be obtained using a GPS attached to the smart node.

On the API, the information can be given using the same method as the edge computing or can be inputted in the online platform, being associated with the specific device.

For the edge computing analysis, the models developed and explained in Section 5.1 were used, after being adapted to run on microcontrollers. The analysis will be done directly on the device without having to send any message to start the analysis or receive the result.

For cloud computing, the analysis is made in the server. For that, a request is made to a specific endpoint, as described in Section 3.5.1, where a script is going to receive the input data, perform the analysis and return the result. Thus, the difference for this case is that the script needs to send a message with the request and wait for a new message to retrieve the data from the JSON array, returning the best protocol from the server.

5.4.2. Decision Process

After knowing how each protocol will behave based on the node characteristics and available protocols, that are given as a result of the previous step, it is needed to decide which protocols to choose.

For edge computing, the logic of the decision to choose the best protocol to use was similar to that used for the Python scripts, presented in Section 3.4.1. In the point-to-point scenario, the script will allow the user to input the various parameters, the device position, the scenario (indoor or outdoor), the number of obstacles and the protocols, that will be retrieved from the IoT node detection mechanism. Then the script will create a matrix of arrays, where each array is composed of that data provided by the user and a transmission power that is associated with each protocol given, based on the information provided in Table 3.1. Then for each array, the RSSI will be predicted and the decision for the best protocol and configuration will be made based on the lowest RSSI value achieved. The script outputs the protocol and transmission power that is associated with the best configuration.

The cloud scenario, in the same fashion as the point-to-point, will allow the user to provide the parameters, the coordinates (Longitude and Latitude) of the device and the protocols. Following the same logic as the previous script, but for this case, the protocols do not have any associated transmission power, so the decision is based only on the predicted RSSI values for the transmission with those parameters, being once again the decision made by choosing the lowest RSSI value. The scripts outputs only the best protocol associated with the given conditions.

These steps were done directly on the device.

For the API these steps were done in the cloud, using Python scripts described in Section 3.4.1, and the best configuration information is retrieved from the JSON array.

5.5. Web Application

The web application and the API are two different ways where it is possible to visualize and analyse the results. After the data is inserted, the processing will happen and the scripts developed will execute the decision process, explained in Section 5.4.2, and the results will be displayed depending on the type of request that was made, either in the web application or in a JSON array.

The Figures 5.5 - 5.11 show how the results are presented in the API or on the online platform.

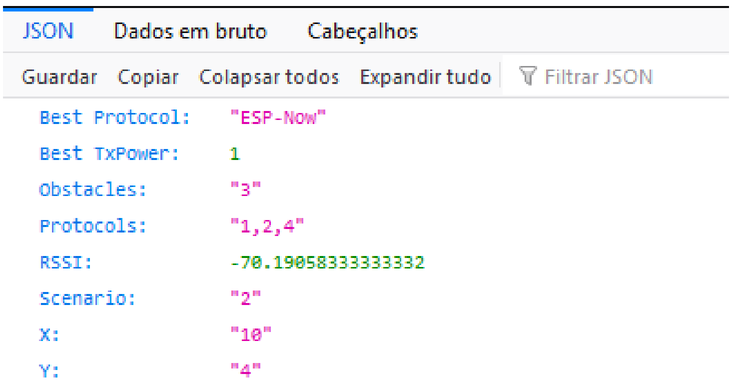

FIGURE 5.5. JSON – Point-to-Point

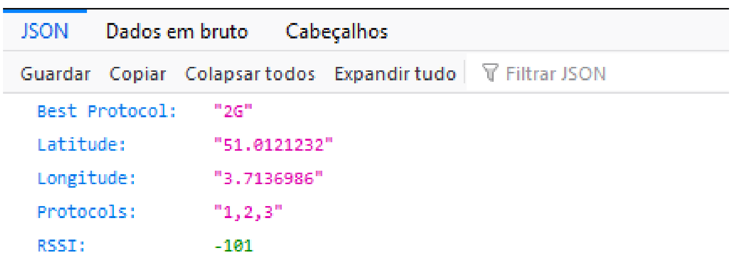

FIGURE 5.6. JSON – Cloud

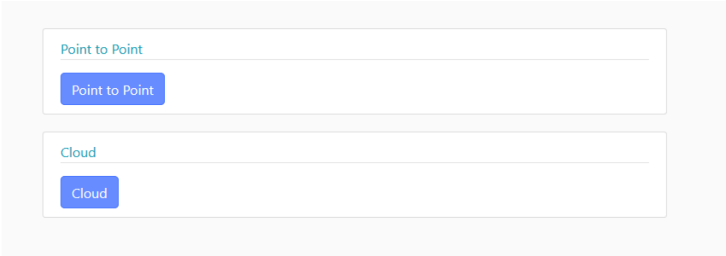

FIGURE 5.7. Main Page of the Online Platform

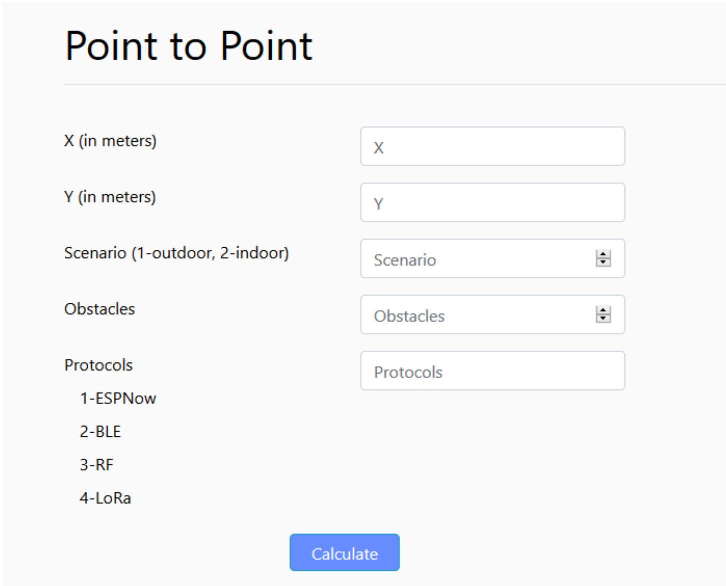

FIGURE 5.8. Page /point_2_point

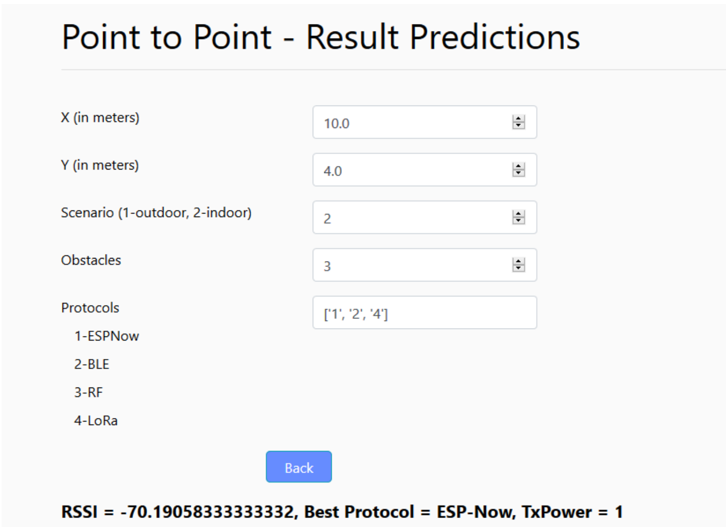

FIGURE 5.9. Page /point_2_point with the Results

5.6. Edge vs Cloud Computing Performance Comparison

To implement this project, not only the ability to create a system capable of creating the best signal quality must be studied, but it is also important to evaluate the decision

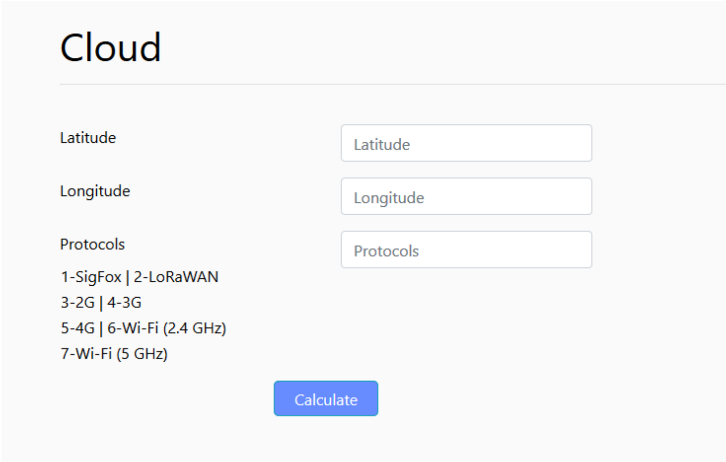

FIGURE 5.10. Page /cloud

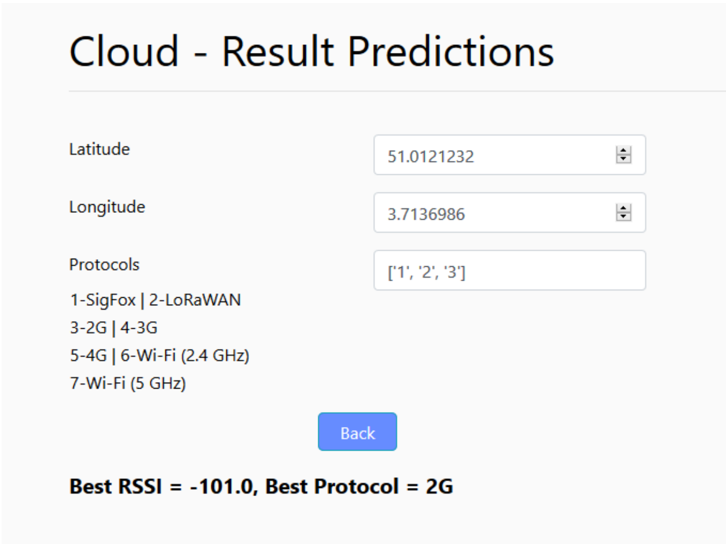

FIGURE 5.11. Page /cloud with the Results

time, the energy used, and the accuracy that the smart node will have, since the idea of this system is also to be a sustainable approach.

To find out if these scenarios are feasible, a set of tests were performed. These tests consisted of measuring the decision time, the energy consumed and accuracy from the time the modules are detected until the decision of which one is best is made.

In the edge computing scenario, as described, all the decision processes and machine learning models are directly implemented in the device. In the cloud computing model, a request to a cloud API is made to obtain the same results.

Table 5.6 shows the obtained results in terms of decision time, energy used and accuracy.

It can be observed that there are big differences from one test to the other, in the case of the API decision time is more than 42 times higher than in edge computing and spends

Table 5.6. Decision Time, Energy Consumption and Accuracy Results

Scenario	**Decision Time (µs)**	**Energy Used (mA)**	**Accuracy (%)**
Edge Computing	10422	37	94
API	440456	114	96.78

3 times more energy, which would be expected, since edge computing makes the request directly to the device but, in the request to the API it is first necessary to connect the Wi-Fi and then to retrieve the request from the API and this takes more time and wastes more energy.

For these tests it is also important to analyse the accuracy of the two scenarios, i.e., to study which of the two, in the API or in the edge computing is obtained in average RSSI values closer to the values that were measured in the tests performed and explained in detail in Section 3.2. Several measurements were taken to get a large sample of RSSI values and then it was calculated how accurate the RSSI is compared to the measured values from the experiment that was performed earlier. It was possible to conclude that both have a very similar and high accuracy, for the edge computing is around 94% and for the API about 96.78%.

Considering all the parameters, edge computing turns out to be the best method to use, because it has a much shorter decision time and uses less energy than API, although it has lower accuracy, it is not much inferior to the accuracy of the API.

CHAPTER 6

Conclusions

6.1. Main Conclusions

In this dissertation, a heterogeneous communication system capable of using several Internet of Things communication protocols was developed, with the help of machine learning algorithms, to be connected to a smart node and to be deployed in any environment. The goal was to create an efficient, reliable and autonomous model that would make the choice of the best protocol to use, without human intervention.

This system has several components, it is composed of a software part and a hardware part. Several scripts were developed to be implemented in a communication shield with the objective of detecting which modules are connected and making the decision of the best configuration to use.

Several steps were taken to be able to achieve the final system. It began by analysing the behaviour of the wireless communication protocols, which includes ESP-Now, Bluetooth Low Energy, Radio Frequency and Long Range, in various circumstances. This test was performed not only to understand how each protocol behaves in different situations, but also to collect data to train the machine learning models. It was possible to conclude that for outdoor scenarios Long Range was the best protocol and for the indoor scenarios the protocol that present better results was ESP-Now.

Using the obtained data, the next step was to analyse it, with the help of machine learning, in order to make predictions of signal quality. To achieve this, several algorithms were studied, to understand which was the best solution for our system and the used data. The machine learning algorithms that were studied were Random Forest, Neural Networks, Decision Tree and Liner Regression and two scenarios were taken into consideration: Point-to-Point, which consists of a direct connection between two devices, that can be distant and have obstacles between them with the objective of evaluating the quality of the signal; and Cloud scenario which aims to create a link between a device and the cloud servers, using a gateway, Wi-Fi router or cellular tower, that might be distant from the device.

After analysing them, it was possible to conclude that the best algorithms, i.e., the ones that managed to obtain the lowest Mean Absolute Error, for the Point-to-Point scenario was the Random Forest which obtained a Mean Absolute Error about 7.44% lower than the second best model, Decision Tree. For the Cloud scenario was the Decision Tree, with a Mean Absolute Error 2.06% smaller than Random Forest, second best. Then these machine learning models were implemented in scripts developed in Python, which will return which is the best protocol to use depending on the environment.

An API has been built that will allow the user to connect to our endpoint and retrieve information about the best communication protocol and configuration for their specifications and scenario.

A web page was also developed that will permit the user to interact and analyse the data in real time. Both the API and the online platform implemented the scripts that were developed in Python in the backend.

Regarding the hardware for this system, a communication shield composed of the ESP32 microcontroller was developed, in a plug&play system with the objective of detecting which radio modules are connected and with the help of edge computing scripts it is possible to know which is the best protocol and the node will self-configure with the best settings.

Several tests were also performed in order to compare the system behaviour in the edge computing scenario and in the API. The decision time, energy consumption and accuracy were evaluated. As expected, edge computing obtained lower values for decision time and energy consumption. API decision time is more than 42 times higher than in edge computing and spends 3 times more energy. However, it was in the API scenario that a better accuracy was obtained, about 96.78%, while edge computing achieves an accuracy of 94%.

It is possible to conclude that the proposed objectives were achieved, an IoT communication system was developed using machine learning techniques, presenting a low-cost solution capable of adapting to its environment that is efficient and sustainable.

6.2. Future Work

Although the system developed is quite complete and the main objectives of this dissertation have been achieved, improvements can be made and some functionalities can be added, both in software and hardware.

Improving the system so that it is possible to use more technologies than are already possible, thus making the system even more complete and even more adaptable to any situation.

A mobile application can also be developed which would make it more convenient for the user to use instead of the web page.

References

[1] S. Li, L. D. Xu, and S. Zhao, "The internet of things: a survey," *Information Systems Frontiers*, vol. 17, no. 2, pp. 243–259, 2015.

[2] F. Wortmann and K. Flüchter, "Internet of Things: Technology and Value Added," *Business and Information Systems Engineering*, vol. 57, no. 3, pp. 221–224, 2015.

[3] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[4] H. Yetgin, K. T. K. Cheung, M. El-Hajjar, and L. Hanzo, "A Survey of Network Lifetime Maximization Techniques in Wireless Sensor Networks," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 2, pp. 828–854, 2017.

[5] H. M. Jawad, R. Nordin, S. K. Gharghan, A. M. Jawad, and M. Ismail, "Energy-efficient wireless sensor networks for precision agriculture: A review," *Sensors (Switzerland)*, vol. 17, no. 8, 2017.

[6] D.-S. Kim and H. Tran-Dang, "An Overview on Wireless Sensor Networks," pp. 101–113, 2019.

[7] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A Survey on Software-Defined Wireless Sensor Networks: Challenges and Design Requirements," *IEEE Access*, vol. 5, pp. 1872–1899, 2017.

[8] C. Saad, B. Mostafa, E. Ahmadi, and H. Abderrahmane, "Comparative Performance Analysis of Wireless Communication Protocols for Intelligent Sensors and Their Applications," *International Journal of Advanced Computer Science and Applications*, vol. 5, no. 4, pp. 76–85, 2014.

[9] J. S. Lee, Y. W. Su, and C. C. Shen, "A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," *IECON Proceedings (Industrial Electronics Conference)*, vol. 4, no. 6, pp. 46–51, 2007.

[10] S. Al-sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, "Review," *2017 8th International Conference on Information Technology (ICIT)*, pp. 685–690, 2017.

[11] T. Rahman and Department, "Provisioning Technical Interoperability within.pdf," *2018 2nd International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, pp. 1–4, 2018.

[12] P. Spachos, I. Papapanagiotou, and K. N. Plataniotis, "Microlocation for Smart Buildings in the Era of the Internet of Things: A Survey of Technologies, Techniques, and Approaches," *IEEE Signal Processing Magazine*, vol. 35, no. 5, pp. 140–152, 2018.

[13] D. Lopez-Perez, A. Garcia-Rodriguez, L. Galati-Giordano, M. Kasslin, and K. Doppler, "IEEE 802.11be Extremely High Throughput: The Next Generation of Wi-Fi Technology beyond 802.11ax," *IEEE Communications Magazine*, vol. 57, no. 9, pp. 113–119, 2019.

[14] E. Khorov, A. Kiryanov, A. Lyakhov, and G. Bianchi, "A tutorial on IEEE 802.11ax high efficiency WLANs," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 197–216, 2019.

[15] E. Ferro and F. Potortì, "Bluetooth and Wi-Fi wireless protocols: A survey and a comparison," *IEEE Wireless Communications*, vol. 12, no. 1, pp. 12–26, 2005.

[16] A. M. Lonzetta, P. Cope, J. Campbell, B. J. Mohd, and T. Hayajneh, "Security vulnerabilities in bluetooth technology as used in IoT," *Journal of Sensor and Actuator Networks*, vol. 7, no. 3, pp. 1–26, 2018.

[17] L. Leonardi, G. Patti, and L. Lo Bello, "Multi-Hop Real-Time Communications over Bluetooth Low Energy Industrial Wireless Mesh Networks," *IEEE Access*, vol. 6, pp. 26 505–26 519, 2018.

[18] D. Čabarkapa, I. Grujić, and P. Pavlović, "Comparative analysis of the Bluetooth Low-Energy indoor positioning systems," *2015 12th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Services, TELSIKS 2015*, pp. 76–79, 2015.

[19] M. Ji, J. Kim, J. Jeon, and Y. Cho, "Analysis of positioning accuracy corresponding to the number of BLE beacons in indoor positioning system," *International Conference on Advanced Communication Technology, ICACT*, vol. 2015-Augus, pp. 92–95, 2015.

[20] "ESP-Now Overview — Espressif Systems," [Online] Available: https://www.espressif.com/en/products/software/esp-now/overview, (visited 06/01/2021).

[21] T. N. Hoang, S. T. Van, and B. D. Nguyen, "ESP-NOW Based Decentralized Low Cost Voice Communication Systems for Buildings," *Proceedings - 2019 International Symposium on Electrical and Electronics Engineering, ISEE 2019*, pp. 108–112, 2019.

[22] H. Malik, H. Pervaiz, M. Mahtab Alam, Y. Le Moullec, A. Kuusik, and M. Ali Imran, "Radio Resource Management Scheme in NB-IoT Systems," *IEEE Access*, vol. 6, pp. 15 051–15 064, 2018.

[23] B. Vejlgaard, M. Lauridsen, H. Nguyen, I. Z. Kovacs, P. Mogensen, and M. Sorensen, "Coverage and Capacity Analysis of Sigfox, LoRa, GPRS, and NB-IoT," *IEEE Vehicular Technology Conference*, vol. 2017-June, 2017.

[24] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "Overview of Cellular LPWAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB-IoT," *2018 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2018*, pp. 197–202, 2018.

[25] R. S. Sinha, Y. Wei, and S. H. Hwang, "A survey on LPWA technology: LoRa and NB-IoT," *ICT Express*, vol. 3, no. 1, pp. 14–21, 2017. [Online]. Available: http://dx.doi.org/10.1016/j.icte.2017.03.004

[26] T. M. Workgroup, "A technical overview of LoRa ® and LoRaWAN ™ What is it?" *Alliance, San Ramon, CA, White Paper*, no. November, 2015. [Online]. Available: https://lora-alliance.org/resource-hub/what-lorawantm

[27] C. Gomez, J. C. Veras, R. Vidal, L. Casals, and J. Paradells, "A sigfox energy consumption model," *Sensors (Switzerland)*, vol. 19, no. 3, 2019.

[28] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 4, pp. 302–3108, 2019.

[29] R. Saravanan and P. Sujatha, "Algorithms : A Perspective of Supervised Learning Approaches in Data Classification," *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, no. Iciccs, pp. 945–949, 2018.

[30] D. Praveen Kumar, T. Amgoth, and C. S. R. Annavarapu, "Machine learning algorithms for wireless sensor networks: A survey," *Information Fusion*, vol. 49, no. April 2018, pp. 1–25, 2019. [Online]. Available: https://doi.org/10.1016/j.inffus.2018.09.013

[31] "1.10. Decision Trees — scikit-learn 0.24.1 documentation," [Online] Available: https://scikit-learn.org/stable/modules/tree.html, (visited 26/02/2021).

[32] "sklearn.ensemble.RandomForestClassifier — scikit-learn 0.24.1 documentation," [Online] Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html, (visited 01/03/2021).

[33] M. Mamdouh, M. A. Elrukhsi, and A. Khattab, "Securing the Internet of Things and Wireless Sensor Networks via Machine Learning: A Survey," *2018 International Conference on Computer and Applications, ICCA 2018*, no. Section II, pp. 215–218, 2018.

[34] "1.17. Neural network models (supervised) — scikit-learn 0.24.1 documentation," [Online] Available: https://scikit-learn.org/stable/modules/neural_networks_supervised.html, (visited 02/03/2021).

[35] "1.4. Support Vector Machines — scikit-learn 0.24.1 documentation," [Online] Available: https://scikit-learn.org/stable/modules/svm.html#, (visited 03/03/2021).

[36] "sklearn.linear_model.LinearRegression — scikit-learn 0.24.1 documentation," [Online] Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html, (visited 05/03/2021).

[37] N. Subramanian and A. Jeyaraj, "Recent security challenges in cloud computing," *Computers and Electrical Engineering*, vol. 71, no. July 2017, pp. 28–42, 2018. [Online]. Available: https://doi.org/10.1016/j.compeleceng.2018.06.006

[38] W. Shi and S. Dustdar, "CLOUD COVER WHY DO WE NEED EDGE COMPUTING? The Promise of Edge Computing," *Computer*, no. 0018, 2016.

[39] "FiPy - Pycom -Five Network Development Board for IoT," [Online] Available: https://pycom.io/product/fipy/?fbclid=IwAR3cNP_auYXB8RM8Lsm6hcgDsHhvcn0Qu5wUEMrb46yoToRHZ4fv5lWcwsk, (visited 08/01/2021).

[40] J. Jagannath, N. Polosky, A. Jagannath, F. Restuccia, and T. Melodia, "Machine learning for wireless communications in the Internet of Things: A comprehensive survey," *Ad Hoc Networks*, vol. 93, 2019.

[41] H. E. Hammouti, M. Ghogho, and S. A. Raza Zaidi, "A Machine Learning Approach to Predicting Coverage in Random Wireless Networks," *2018 IEEE Globecom Workshops, GC Wkshps 2018 - Proceedings*, pp. 1–6, 2019.

[42] A. Gupta and M. Fujinami, "Battery Optimal Configuration of Transmission Settings in LoRa Moving Nodes," *2019 16th IEEE Annual Consumer Communications and Networking Conference, CCNC 2019*, pp. 1–6, 2019.

[43] H. Yan and H. Hu, “Study on Energy Saving Algorithm of LoRa Terminal Based on Neural Network,” *Proceedings - 2018 3rd International Conference on Smart City and Systems Engineering, ICSCSE 2018*, pp. 908–911, 2018.

[44] R. Li, X. Li, and Y. DIng, “Link Prediction Algorithm for BLE Mesh Network in Health Monitoring System,” *Proceedings of the 32nd Chinese Control and Decision Conference, CCDC 2020*, pp. 1997–2001, 2020.

[45] “ESP32 Wi-Fi & Bluetooth MCU I Espressif Systems,” [Online] Available: https://www.espressif.com/en/products/socs/esp32, (visited 07/01/2021).

[46] A. Maier, A. Sharp, and Y. Vagapov, “Comparative analysis and practical implementation of the ESP32 microcontroller module for the internet of things,” *2017 Internet Technologies and Applications, ITA 2017 - Proceedings of the 7th International Conference*, pp. 143–148, 2017.

[47] “scikit-learn: machine learning in Python — scikit-learn 0.24.2 documentation,” [Online] Available: https://scikit-learn.org/stable/, (visited 01/08/2021).

[48] “sklearn.model_selection.RandomizedSearchCV — scikit-learn 0.24.2 documentation,” [Online] Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html, (visited 01/08/2021).

[49] P. Chawdhry, G. Folloni, S. Luzardi, and S. Lumachi, “European Cellular signal strength coverage,” *[Dataset] European Commission, Joint Research Centre (JRC).*, 2016. [Online]. Available: http://data.europa.eu/89h/jrc-netbravo-netbravo-od-eu-cellular

[50] M. Aernouts, R. Berkvens, K. Van Vlaenderen, and M. Weyn, “Sigfox and LoRaWAN Datasets for Fingerprint Localization in Large Urban and Rural Areas (Version 1.3),,” *[Dataset] Zenodo*, 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3904158

[51] D. F. Fernandes, A. Raimundo, F. Cercas, P. J. Sebastiao, R. Dinis, and L. S. Ferreira, “Comparison of Artificial Intelligence and Semi-Empirical Methodologies for Estimation of Coverage in Mobile Networks,” *IEEE Access*, vol. 8, pp. 139 803–139 812, 2020.

Appendices

APPENDIX A

Scientific Contributions

Prediction of Link Quality for IoT Cloud Communications supported by Machine Learning

Beatriz Dias
Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal
bcdds@iscte-iul.pt

André Glória
Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal
afxga@iscte-iul.pt

Pedro Sebastião
Instituto Universitário de Lisboa (ISCTE-IUL)
Lisbon, Portugal
pedro.sebastiao@iscte-iul.pt

***Abstract*—This paper introduces an evaluation of machine learning techniques used to predict the link quality of communications done by IoT nodes, based on regression models. The proposed methodology is able to predict the link quality of the most typical cloud communication protocols, such as cellular, Wi-Fi, SigFox and LoRaWAN, based on the node location. To discover the best model to achieve this, a set of machine learning techniques were implemented, including Linear Regression, Decision Tree, Random Forest and Neural Networks, being the results compared. Results showed that Decisions Trees achieve the best efficiency, with a margin of error of 7.172 dBm, after cross-validation. The followed methodology, its implementation and experimental results are detailed in this paper.**

***Index Terms*—Machine Learning, Wireless Communications, Internet of Things, Regressions**

I. Introduction

Nowadays a major role in our life's is defined by technology and we no longer can live without them. Right now, we already live in a smart world and, without a doubt, it will not stop here. The Internet of Things (IoT), responsible for connecting objects to the internet and creating smart objects, will continue to grow, and to achieve its main goal of improve human activities and experiences, IoT devices need to be more efficient when the user need them.

However, with all this growth, communication systems will become more complex and problems such as latency and congestion will appear. It will become more difficult for a sensor network to be able to work in any scenario, because in many occasions there may also be a lack of coverage or inefficiency of communication protocols in certain environments, especially comparing indoor and outdoor environments, or cities and rural areas. With 50 billions devices expected to be connected by 2030 [1], communications will take a major role in allowing them to work in a network, exchange messages among them and with the cloud and keep everything up and running.

With the rise of IoT, standard communication protocols, such as Wi-Fi, Bluetooth or cellular, needed to be updated. New standard such as Narrow Band IoT (NB-IoT), a cellular protocol design to send small packages of data, or Bluetooth Low Energy (BLE), an adaptation of Bluetooth for IoT devices, start to be part of every IoT project. But as devices started to be more outdoor than indoor, and connectivity went from small homes to entire cities, new protocols such as LoRaWAN or Sigfox, both in the Low-Power Wide-Area Networks (LPWAN), started to be the go to choice for IoT systems.

With this new reality, the problem IoT connectivity goes from which protocol is best for my system to which protocol is available and ready to use at this moment. As nodes start to have more computational power and hardware capabilities, they are able to carry more than one communication module and to have some intelligence to decide which to use at a certain moment, creating heterogeneous communication schemes.

This creates a new problem, as this intelligence needs to be done in real-time and autonomously, it has become impossible to analyze the data manually and therefore it was necessary to have technologies with the ability to interpret and process this data with human intervention, and this is how Machine Learning (ML) arises.

ML focus on creating knowledge from data and use that knowledge to improve the accuracy and efficiency of applications without human intervention. The algorithms are 'trained' to inspect data and search for patterns in order to achieve better decisions, combining data with statistical tools to predict an output. The idea is that the systems learn automatically, with as minimal human intervention as possible [2], [3].

Machine Learning and IoT are two fields that are more and more dependent on each other, and communication is one the main areas that benefit from this. With several studies conducted using machine learning techniques in wireless communication, with security, anomaly detections, improving communication efficiency and predicting energy consumption being the main areas of research, it can be applied to anything from traffic light control, to smart farm, health care or industrial IoT.

This paper presents a study of Machine Learning Regression algorithms, where Random Forest (RF), Neural Networks (NN), Decision Trees (DT) and Linear Regression (LR) will be studied, in order to build a system capable of predicting the link quality of the major IoT cloud communication system,

This work was supported in part by ISCTE - Instituto Universitário de Lisboa from Portugal under the project ISCTE-IUL-ISTA-BM-2018

based on the position of the node.

After this introduction, some research found in the literature is presented, being followed by an introduction to the Machine Learning techniques. Then, the methodology for the study is presented, including the used dataset and training process. Results and discussion based on the following methodology are given, with the conclusions from our research being presented at the end.

II. Related Work

With the great growth of Machine Learning associated with wireless communications, several researches can be found in the literature. A great survey of this research spectrum can be found in [4], covering several applications from security, interference, link configuration, node locations and several others.

Our goal of predicting the link quality based on node location and the type of protocol used, in order to create a heterogeneous communication scheme that switches protocols as a better one is available, falls inside this research. Although, almost none similar approaches were found in the literature, being mainly link or coverage predictions for specific protocols found.

In [5], the main objective is to perform a coverage prediction in wireless sensor networks using Machine Learning, to determine an accurate mapping between network features and network performance. It was concluded that the Neural Networks model with three layers was sufficient to achieve high accuracy, and with more than three layers the accuracy did not increase significantly.

The literature also shows some work being done to create a better LoRa link using Machine Learning techniques, with [6], using Dynamic Selection, with 96% efficiency, and [7] using Neural Networks to improve the energy efficiency of LoRa connection, with a 99.92% accuracy, but only with 200 samples. Some works were also found with the use of machine learning to predict the link quality of BLE mesh networks [8].

As explained, several studies using Machine Learning to predict communications quality can be found in the literature. Nevertheless the proposed solution in this paper, as far as we know, has never been developed before, so our approach presents innovation.

III. Machine Learning Techniques

Machine Learning has various learning techniques that can be used, with the most common being supervised learning and unsupervised learning. For supervised learning, a pair of input-output training data is provided with this technique analyzing the data and the relationship between them and in the end, being possible to predict a function from the input with the best estimation of output [9]. A supervised learning algorithm learns from labeled training data, helping to predict outcomes for unforeseen data.

Supervised learning can be divided into regression and classification [9]. In regression, it is predicted as a dependent variable Y based on the input independent variable X through statistical processes that estimate the relationship between the variables.

Inside supervised learning there are multiple techniques that can be used to do regressions, being the most common described next.

1) Random Forest (RF) uses a set of decision trees to make regressions. To each each tree is given a sample, being each built from a portion of the dataset, then each tree will give a classification and finally depending on the results of the combination of all trees the algorithm evaluates which is the best option [10]–[12].
2) Neural Network (NN) is an algorithm composed of many neurons connected to each other, similar to a human nervous system. This method is composed of three layers, input, hidden and output. Each neuron analyzes a part of the input received and sends the information to the neuron in the next layer. The process ends when a final output is found. NN can be used to solve nonlinear and complex problems. For the implementation of this technology, a Multi-layer Perceptron (MLP) was used. MLP is a supervised learning algorithm that learns a function $f(.) : R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output [13], [14].
3) Decision Trees (DT) are tree-based algorithms that can be used for classification and regression. This method is established on a set of “if-then” rules to enhance readability. The data is split using a hierarchical partition, being this split done iteratively. A decision tree contains two different nodes, the decision nodes, where the decision is made among the alternatives, and leaf nodes, which are the final outcome is chosen. Decision Trees are simple to understand and to interpret and also reduces ambiguity in decision-making [3], [13].
4) Linear Regression (LR) allows for the model to create a hyperplane based on the input data, being this line able to predict the data based on the inputs. To calculate this line, previous data is used, being then used to predict the results when new data is used [**?**].

IV. Regression Model

As said before, our methodology uses a regression model for the computation analysis. This methodology depends on the models being trained with know data, allowing the model to be configured and achieve the lowest margin of error possible. The next sections explain, in detail, how this process was done and the obtained model results.

The goal of this model is to receive as input the location of the node (Latitude and Longitude) and the communication protocol to use and output the predicted quality link.

A. Dataset

For a machine learning model to work, it needs to be trained with a set of supervised data, that include the output desired for a set of parameters. It is with these data, that the model will learn and predict upon future data.

The used dataset is composed of crowd-sourced data collected by the NetBravo project [15], a crowd-sourcing project, by the European Union, designed to gather and share wireless communication data about cellular coverage, Wi-Fi channel occupancy and broadband. The dataset contains records of the characteristics of the signal for Wi-Fi, 2G, 3G or 4G. That dataset was combined with LoRaWAN and SigFox data from a fingerprint localization dataset for large outdoor environments [16], that contains information about the base station, delivery time and the Received Signal Strength Indicator (RSSI) per base station.

This allows us to have a dataset of multiple cloud wireless protocols such as 2G, 3G, 4G, Wi-Fi, LoRaWAN and Sigfox. The NetBravo dataset contains points from across the entire European Union countries, whereas the other dataset only contains data from Antwerp, Belgium. As such, to have a more reliable result, only the NetBravo entries from the Antwerp region were considered in the final dataset. With that in mind, the final dataset is composed of 849517 entries, with the following parameters:

- Latitude – Latitude of the node;
- Longitude – Longitude of the node;
- Protocol – Protocol used in the data transimssion: 1 - SigFox, 2 - LoRaWAN, 3 - 2G, 4 - 3G, 5 - 4G, 6 - Wi-Fi (2.4GHz), 7 - Wi-Fi (5GHz);
- RSSI – Link quality of the data transmission;

B. Training Methodology

In order to train the regression model, the following steps were done, using Python, the scikit-learn libraries [17] and the Anaconda environment.

1) Each model was trained using the presented dataset and the default configuration presented in their documentation. A such, it is easier to compare the performance of each model, in terms of margin of error, and understand which are capable of being improved o achieve better results.
2) After testing the default configuration, each model undergoes into a hyper parametrization tuning, that compares multiple configurations of the same model, in order to assess which obtains the best performance, facing the dataset and the goal. For that, a RandomizedSearchCV was used [18];
3) In the final step, and to guarantee the validity of the model, the best configuration of the model is submitted into a Stratified K-Fold cross validation. This allows to understand if the model is under or over fitted. Using a set of five folds, each using a different part of the training dataset, allowing for the model to check on every single datapoint, it is possible to validate the real model performance, as each of the folds will create a different outcome, that are averaged, allowing for a better perception of the error margin and variation, as more data is used to fit the model;

V. Results & Discussion

To obtain the best regression model capable of predicting the link quality of data transmissions, based on node position, the presented methodology was used.

The used dataset was divided into a training, validation and testing dataset. These were divided as follows: 70% for training, 20% for validation and 10% for testing. To evaluate the model performance, and since a regression is used, the Mean Absolute Error (MAE) metric will be used, as it is the most common metric for regression. The MAE value indicates the average absolute error between the original data and the predicted value, using (1) [19], where P_{rx} is the original value, $\hat{P}_{rx}$ is the predicted value, and N the number of samples.

$$MAE[dBm] = \frac{1}{N} \cdot \sum_{i=1}^{N} |P_{rx_i} - \hat{P}_{rx_i}| \qquad (1)$$

When the MAE is closer to 0, it indicates that the model matches the original data.

Figure 1 presents the obtained results for each step of the methodology for predicting the link quality.

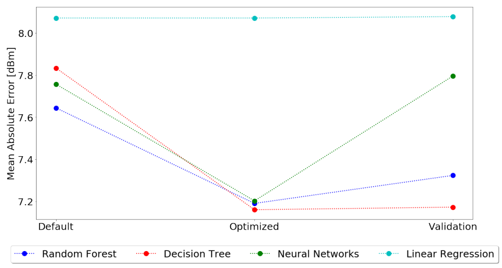

Fig. 1. Regression Model Results

As is possible to see, the values have a noticeable variation depending on the type of test. Analyzing the graph, it can be seen that the hyper parametrization values are the ones that always have the lowest MAE values, then the cross-validation and finally the default values with the highest values, except in the scenario where the Neural Network algorithm was used, where the cross-validation has the value slightly higher than the default value. This follows the expected behavior due to the presented methodology, since after the first test with the default values, these values will be tuned to improve, being predictable that in the optimized scenario better results will be obtained. Then, in the cross-validation multi combinations of the dataset are tested and the MAE for each fold is averaged, thus it is expected that the MAE increases.

Comparing the results obtained across the models, it is possible to conclude that the best algorithm is the Decision Tree, with a margin of error of 7.172 dBm, after the cross-validation phase. When compared with Random Forest, which is the second best model, it achieves a lower MAE by 0.0301 dBm and 0.151 dBm, in the optimized and cross-validation phases, respectively. As said, Neural Networks have a cross-validation MAE higher than the default and optimized stage, which means that the model does not work well with unseen data. Regarding Linear Regression, as expected, presented the worst results among all models.

In terms of network performance, according to [19] a good propagation is reached is the model had a margin of error lower than 9 dBm, so Random Forest is able to create a good prediction model for link quality in a communication network.

VI. Conclusions

The goal of this paper was to develop a study of some of the most common algorithms in Machine Learning to predict the quality of a signal using different protocols, based on the coordinates of the device.

It started by studying some related work, finding some research of Machine Learning in wireless communication scenarios, although none was found that tackles the prediction of communication link for heterogeneous communication schemes.

A detailed study was made of the Machine Learning techniques and the algorithms used, these being Random Forest, Neural Networks, Decision Tree and Linear Regression. A methodology was followed to do this study, starting with using the default configuration values, then performing hyper parameterization of these values, and finally a cross-validation.

After a detailed study of the mean absolute error (MAE) values of all the model under different scenarios, it was possible to conclude that the Decision Tree algorithm had the best results, with the results on the optimized and cross validation stage having the lowest MAE. Random Forest achieved close results, being slightly worse than DT by 0.0301 dBm and 0.151 dBm, in the optimized and validation stage. It was also possible to conclude that Neural Networks did not perform well when unseen data was introduced in the model, as the cross validation results were worse than the default ones. Linear Regressions, as expected, had the worst results. It is then possible to conclude that the Decision Tree algorithm, that obtained the lowest margin of error, is the best solution for a methodology capable of predicting the link quality based on node location.

With the best model selected, the next steps in our research are to develop an algorithm that implements this model and automatically decides the best protocol each time new data needs to be transmitted and then implement that solution in a device and test it in a real case scenario.

References

[1] R. P. Kumar and D. S. Smys, "A Novel Report on Architecture, Protocols and Applications in Internet of Things (IoT)," in *2018 2nd International Conference on Inventive Systems and Control (ICISC)*, no. Icisc, pp. 1156–1161, IEEE, 2018.

[2] Y. Sun, M. Peng, Y. Zhou, Y. Huang, and S. Mao, "Application of Machine Learning in Wireless Networks: Key Techniques and Open Issues," *IEEE Communications Surveys & Tutorials*, vol. 21, pp. 3072–3108, 9 2018.

[3] R. Saravanan and P. Sujatha, "A State of Art Techniques on Machine Learning Algorithms: A Perspective of Supervised Learning Approaches in Data Classification," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 945–949, 2018.

[4] J. Jagannath, N. Polosky, A. Jagannath, F. Restuccia, and T. Melodia, "Machine Learning for Wireless Communications in the Internet of Things: A Comprehensive Survey," 1 2019.

[5] H. E. Hammouti, M. Ghogho, and S. A. R. Zaid, "A Machine Learning Approach to Predicting Coverage in Random Wireless Networks," in *2018 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, 2018.

[6] A. Gupta and M. Fujinami, "Battery Optimal Configuration of Transmission Settings in LoRa Moving Nodes," in *2019 16th IEEE Annual Consumer Communications and Networking Conference, CCNC 2019*, Institute of Electrical and Electronics Engineers Inc., 2 2019.

[7] H. Yan and H. Hu, "Study on Energy Saving Algorithm of LoRa Terminal Based on Neural Network," in *Proceedings - 2018 3rd International Conference on Smart City and Systems Engineering, ICSCSE 2018*, pp. 908–911, Institute of Electrical and Electronics Engineers Inc., 5 2019.

[8] R. Li, X. Li, and Y. Ding, "Link Prediction Algorithm for BLE Mesh Network in Health Monitoring System," in *32nd Chinese Control and Decision Conference (CCDC 2020)*, pp. 1997–2001, 2020.

[9] D. Praveen Kumar, T. Amgoth, and C. S. R. Annavarapu, "Machine learning algorithms for wireless sensor networks: A survey," *Information Fusion*, vol. 49, pp. 1–25, 9 2019.

[10] H. Luo, X. Pan, Q. Wang, S. Ye, and Y. Qian, "Logistic regression and random forest for effective imbalanced classification," in *Proceedings - International Computer Software and Applications Conference*, vol. 1, pp. 916–917, IEEE Computer Society, 7 2019.

[11] M. Kayri and I. Kayri, "The Performance Comparison of Multiple Linear Regression, Random Forest and Artificial Neural Network by using Photovoltaic and Atmospheric Data," in *2017 14th International Conference on Engineering of Modern Electric Sy stems (EMES)*, 2017.

[12] J. K. Jaiswal and R. Samikannu, "Application of Random Forest Algorithm on Feature Subset Selection and Classification and Regression," in *2nd World Congress on Computing and Communication Technologies, WCCCT 2017*, pp. 65–68, Institute of Electrical and Electronics Engineers Inc., 10 2017.

[13] M. Mamdouh, M. A. Elrukhsi, and A. Khattab, "Securing the Internet of Things and Wireless Sensor Networks via Machine Learning: A Survey," in *2018 International Conference on Computer and Applications, ICCA 2018*, pp. 215–218, Institute of Electrical and Electronics Engineers Inc., 9 2018.

[14] T. Hastie, R. Tibshirani, and J. Friedman, *Springer Series in Statistics The Elements of Statistical Learning Data Mining, Inference, and Prediction*. Springer, 2nd ed., 2008.

[15] P. Chawdhry, G. Folloni, S. Luzardi, and S. Lumachi, "European Cellular signal strength coverage," 2016. [Dataset] European Commission, Joint Research Centre (JRC). http://data.europa.eu/89h/jrc-netbravo-netbravo-od-eu-cellular.

[16] M. Aernouts, R. Berkvens, K. Van Vlaenderen, and M. Weyn, "Sigfox and LoRaWAN Datasets for Fingerprint Localization in Large Urban and Rural Areas (Version 1.3)," 2019. [Dataset] Zenodo. https://doi.org/10.5281/zenodo.3904158.

[17] scikit-learn, "scikit-learn," 2021. [Online] Available: https://scikit-learn.org/stable/, (visited 16/02/2021).

[18] scikit-learn, "RandomizedSearchCV," 2021. [Online] Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html, (visited 16/02/2021).

[19] D. F. Fernandes, A. Raimundo, F. Cercas, P. J. Sebastiao, R. Dinis, and L. S. Ferreira, "Comparison of Artificial Intelligence and Semi-Empirical Methodologies for Estimation of Coverage in Mobile Networks," *IEEE Access*, vol. 8, pp. 139803–139812, 2020.