INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Sustainable Modular IoT Solution for Smart Cities Applications Supported By Machine Learning Algorithms

André Filipe Xavier da Glória

PhD in Information Science and Technology

Supervisor:
Doctor Pedro Joaquim Amaro Sebastião, Assistant Professor,
ISCTE – Instituto Universitário de Lisboa

August, 2021

TECNOLOGIAS
E ARQUITETURA

Department of Information Science and Technology

Sustainable Modular IoT Solution for Smart Cities Applications Supported By Machine Learning Algorithms

André Filipe Xavier da Glória

PhD in Information Science and Technology

Jury:
Doctor João Carlos Amaro Ferreira, Assistant Professor with Aggregation, ISCTE – Instituto Universitário de Lisboa (President)
Doctor José Alberto Gouveia Fonseca, Associate Professor, Universidade de Aveiro
Doctor Vítor Manuel Rodrigues Viegas, Assistant Professor, Escola Naval
Doctor Francisco António Bucho Cercas, Full Professor, ISCTE – Instituto Universitário de Lisboa
Doctor Pedro Joaquim Amaro Sebastião, Assistant Professor, ISCTE - Instituto Universitário de Lisboa

August, 2021

Acknowledgment

First of all, I would like to acknowledge Prof. Dr. Pedro Sebastião, not only for accepting to be my supervisor, but also for all the help given throughout the time, not only in academic but also in professional and personal matters. This journey of thinking of doing a Ph.D. and now reaching that goal was not possible without him, without his inspiration, help, knowledge, dedication and friendship.

To Prof. Dr. Nuno Garrido and Prof. Dr. Octavian Postolache, for being part of my research monitoring committee, that throughout these years helped me evaluate the work I developed and steer me into the right direction when needed.

To ISCTE, for providing me all the resources needed to do my studies and research and for being a second home for the last 8 years, in which I studied, organized several events, and had the privilege to teach over four semesters. A word of appreciation also to all the staff with whom I cross paths along these years, mainly to Marisa Manteigas and Fátima Estevens, for helping me with all the bureaucracies needed, and to the Evaluation Committee, composed by Prof. Dr. Bráulio Alturas, Prof. Dr. Rui Lopes and Prof. Dr. Octavian Postolache, and to ISTA, for believing in my research and awarded me, during these three years, the ISTA 3rd Cycle Merit Scholarship.

To my family, especially to my parents, Sandra and Paulo, for always supporting my studies, for encouraging me to have the education they were not able to get and that always done the possible and the impossible for providing me the opportunity to reach my goals. Without them it was not possible to get all these achievements and for that I will forever be grateful.

To my good friends André Marques and Daniel Fernandes, who I met in my first year at ISCTE and that for the past 8 years accompanied me in the various challenges that brought me to where I am today. To Inês and Filipa Gomes, which without this PhD I would never met or have the friendship that we have today, and to whom I appreciate for all the talks, walks and food breaks, they were all essential for me to finish my work; and to Ana Marta Contente, for always being there to support me and for having a funny story or stupid thing to tell me when I most needed. A special thanks to my friends

Carolina Dionisio, Gonçalo Simões, João Cardoso, João Alves Coelho and Maria Ines Pires for accepting my challenge to develop their master thesis under my supervision and alongside my research. To Patricia Santos, António Raimundo, Catarina Duque, Mariana Cardoso, João Antão and to all my other friends and colleagues, that in some way helped me throughout these three years.

Lastly, and since there is always that one person that influences you to reach for the stars, an exceptional thanks for my 1st grade Professor Fandy. Thanks for teaching that little kid that used to bring toy cars to play under his desk during class time how to read and do maths, and for always reminding me not to give up and that grades do not define who you are. Although you are not here to see it, I know that you would be proud of the outcome.

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

Resumo

A Internet das Coisas (IoT) e as Cidades Inteligentes são hoje uma grande tendência, mas com a rápida evolução destes sistemas são vários os desafios que põem em causa a sua aceitação por parte das populações, maioritariamente devido a problemas ambientais e de sustentabilidade. Esta Tese introduz um novo sistema composto por nós de IoT inteligentes que são auto-configuráveis e sustentáveis suportados por de aprendizagem automática, e o trabalho de investigação e desenvolvimento para se obter uma solução inovadora que considera a análise de dados, comunicações sem fios e o desenvolvimento do hardware e software. Para todos estes, os conceitos chave são introduzidos, as metodologias de investigação, testes e resultados são apresentados e discutidos, bem como todo o desenvolvimento e implementação. Através do trabalho desenvolvido mostra-se que as Árvores Aleatórias são a melhor escolha para análise de dados em termos da auto-configuração do hardware e sistema de comunicações e que a computação nos nós tem uma vantagem em termos de eficiência energética e latência. O sistema de configuração autónoma de comunicações foi capaz de criar um nós 65% mais sustentável, em termos energéticos, comprometendo apenas em 13% a qualidade do serviço. A solução modular do nó inteligente apresentou vantagens na integração, escalabilidade e implementação de projectos para Cidades Inteligentes quando comparado com soluções tradicionais, reduzindo em 45% o consumo energético e 60% a troca de mensagens, sem comprometer a qualidade do sistema. A implementação deste novo sistema irá ajudar as cidades inteligentes, em todo o mundo, a diminuir os seus problemas ambientais e a cumprir com as normas e regulamentos para reduzir as emissões de CO_2.

Palavras-Chave: Internet das Coisas, Aprendizagem Automática, Cidades Inteligentes, Comunicações Sem Fios, Sustentabilidade.

Abstract

The Internet of Things (IoT) and Smart Cities are nowadays a big trend, but with the proliferation of these systems several challenges start to appear and put in jeopardy the acceptance by the population, mainly in terms of sustainability and environmental issues. This Thesis introduces a new system composed by a modular IoT smart node that is self-configurable and sustainable with the support of machine learning techniques, as well as the research and development to achieve a innovative solution considering data analysis, wireless communications and hardware and software development. For all these, concepts are introduced, research methodologies, tests and results are presented and discussed as well as the development and implementation. The developed research and methodology shows that Random Forest was the best choice for the data analysis in the self-configuration of the hardware and communication systems and that Edge Computing has an advantage in terms of energy efficiency and latency. The autonomous communication system was able to create a 65% more sustainable node, in terms of energy consumption, with only a 13% decrease in quality of service. The modular approach for the smart node presented advantages in the integration, scalability and implementation of smart cities projects when facing traditional implementations, reducing up to 45% the energy consumption of the overall system and 60% of messages exchanged, without compromising the system performance. The deployment of this new system will help Smart Cities, in a worldwide fashion, to decrease their environmental issues and comply with rules and regulations to reduce CO_2 emission.

Keywords: Internet of Things, Machine Learning, Smart Cities, Wireless Communications, Sustainability.

Contents

List of Figures

List of Tables

List of Acronyms

AI: Artificial Intelligence.

BLE: Bluetooth Low Energy.
BLM: Best Link Model.

CC: Cloud Computing.
CSS: Chirp Spread Spectrum.

D-BPSK: Differential Binary Phase-Shift Keying.
D2D: Device to Device.
D2S: Device to Server.
DT: Decision Tree.

EC: Edge Computing.
EFM: Energy Efficiency Model.

HVAC: Heating, Ventilation and Air Conditioning.

ICT: Information and Communication Technologies.
IoT: Internet of Things.

LoRa: Long Range.
LPWAN: Low-Power Wide-Area Networks.
LR: Linear Regression.

ML: Machine Learning.
MLP: Multilayer Perceptron.
MQTT: Message Queuing Telemetry Transport.

NB-IoT: NarrowBand IoT.

NN: Neural Networks.

PCB: Printed Circuit Board.

RF: Random Forest.
RLM: Reliable Link Model.
RSSI: Received Signal Strength Indicator.

S2S: Server to Server.
SC: Smart Cities.
SVM: Support Vector Machines.

UNB: Ultra-Narrow Band.

Wi-Fi: Wireless Fidelity.
WSAN: Wireless Sensor and Actor Network.
WSN: Wireless Sensor Network.

CHAPTER 1

Introduction

As S. Routray and P. Sharmila [**1**] said "Green technologies and energy efficient process in Internet of Things (IoT) are well researched areas in which optimization of resources takes the central position". IoT is impacting the world by creating new markets and innovation, as it connects a wide range of networks in both economy and society, requiring innovation in information, communications and regulations [**2**].

In a world where natural resources, such as water, are starting to vanish and material goods prices, like energy, are at all time highs, the need to create more efficient processes is a must in order to improve sustainability. To achieve this, innovations on a large scale are required and IoT can take a major role. Combining IoT, sustainability and Machine Learning (ML) algorithms is possible to achieve a disruptive innovation capable of saving energy, decreasing CO_2 production, minimizing costs, fewer fuels, waste reduction and time saving.

These technologies are affecting the way we live, work and play, and one of the opportunities that is rising is the concept of Smart Cities (SC), where thousands of sensors are being deployed and generating massive amounts of data that is analysed to enable city services to be more responsive to the needs of the citizens [**3**]. Nowadays, cities face a variety of challenges, from job creation, economic growth, social resilience, but mainly environmental sustainability. With cities representing 2% of the Earth's surface and over 50% of the Earth population residing in urban areas, with an expected growth to over 60% by 2050, [**4, 5, 6**] urbanization has become a major characteristics of economics and social development. Not only will this contribute for a consumption of about three-quarter of city resources [**7, 5**], it will also contribute to the release of more than 80% of CO_2 emissions [**6**]. All of these means that managing urban areas has become one of the most critical challenges our society faces today.

"Green Tech" appears as a need to reduce the carbon footprint, waste and to create a more efficient energy consumption in business and day-by-day procedures, in order to improve sustainability [**2, 8**]. To achieve this, hardware and software need to embrace this new requires where, in hardware, devices must consume less energy without compromising

their effectiveness and efficiency, as in software, more efficient algorithms must take into consideration using the minimum resources required to achieve the goal while consuming less energy [**9**].

Sustainability is nowadays directly linked to innovation since all around the world people are becoming more concerned with climate changes, water shortage, clean energies and other challenges that require the development of new products, business and services that can lead to advances not only in environmental dimensions but also in social and economics. New technologies such as IoT are impacting the world, with a huge research being done in the areas of sustainability, where a wide innovation is being seen in the approach of sustainability challenges [**2**]. Creating sustainability using rising technologies allows devices, objects and processes to be more reliable, tough, autonomous and smart.

On the other hand, the proliferation of these technologies and services are creating a new set of challenges and questions that are affecting the sustainability and the security of cities, mainly:

- Where is the data coming from and can it be trusted?
- With so many manufacturers, what are the standards for these solutions?
- Are the autonomous learning and decision systems doing the right thing?
- Is our privacy and security at risk?
- How to power all these devices?
- Are these solutions scalable?
- Where is all this data being stored?
- Is the amount of new devices and communications affecting our life's?
- What is the environmental impact of these solutions?

These problems, that come from the rapid proliferation of IoT and Smart Cities, are still creating some resistance and mistrust in these technologies, leading to some part of the population avoiding their acceptance.

1.1. Problem Statement

This doctoral programme consists, as its main goal, to introduce a new way of creating generalized Wireless Sensor Network (WSN) capable of performing in distinct fields, such as houses, pools, and agriculture, without the need to major hardware and software changes and also adaptable without human intervention, using Artificial Intelligence (AI) algorithms, to achieve a better efficiency in process that can lead to savings for the final

user, not only monetary but also in natural resources, such as water and energy, leading to a more sustainable environment.

In order to deliver this, an adaptable IoT node will be developed, capable of performing in almost every specification or environment, including an autonomous communication scheme, auto-sustainable and modular in terms of sensors and actuators. Also, a cross platform dashboard will be developed to serve as a gateway between the client and the network. This platform will gather information acquired by the WSN, storing the information on a cloud-based server, analyzing and processing it and then create more efficient processes. By recurring to AI algorithms, the network will be able to adapt itself, in an autonomous way, to create better decisions, not only in the nodes features, but also in the action to perform facing the sensor data. For example, in an irrigation system, determine the correct amount of water to use taking into consideration the soil moisture, type of plants, overall weather forecast and other important input parameters, to obtain the best automatized and sustainable (economic, social and environment) decision. By joining these presented solutions, it will be possible to create an innovative service and a product that can improve not only the consumption of natural and material goods but also the user life. When compared with traditional methods, the research and development to be considered in this doctoral program intends to obtain an affordable and efficient solution that can be transferred to the market.

Our research, not being able to tackle every issue presented, will focus mainly on the heterogeneous solutions for the Smart City context, sustainability, battery life and easy scalability, as well as the development of new learning systems.

Hence, this research project was divided into three modules, that when integrated and synchronized create a system capable of fulfilling the proposed goals. The modules are as follows:

(1) The development of a fully modular and adaptable Wireless Sensor Network, composed of a set of smart nodes that can adapt to any specification without needing human intervention, in either software or hardware. This will tackle the heterogeneous problem as a one-fits-all solution, instead of the current situation, where new hardware is developed for each specification. Also, the scalability and battery capacity will be tackled, as with, an adaptable network, the deployment of new solution will be simplified and done in a more sustainable way;

(2) The development of an autonomous configuration system for communications, capable of tackling both device-to-device and device-to-server communication, that will allow our WSN to be more adaptable, as it can use the best protocol available in the installation site, improving the interoperability and availability regardless of geography. This will not only improve our adaptability, since the nodes will be able to communicate with the best solution every time, even if condition change over time, but also will improve battery life of our devices and scalability, as the installation and configuration is done autonomously;

(3) The development of the learning system, using Machine Learning algorithms, in order to analyze, in real time, the data collected by the WSN and support all the previous modules, improving, automatically, the system to be more effective, efficient and reliable.

1.2. Thesis Structure and Contribution of Research

In this introductory section, we provide an overview of the problem addressed in this thesis, as well as the thesis structure and scientific publications produced over the past three years. Other research contributions are also presented.

In Chapter 2, we review the current state of the art in the field of IoT, Smart Cities and how the proliferation of these is causing problems in the sustainability and general public awareness and acceptance of these new systems. This chapter is divided into four main sections: an introduction to IoT and Smart Cities and how they evolved in the last few years; the enabling technologies associated with them, from devices to communication and computing techniques; the main problems rising from this rapid proliferation; and how the environment is being affected and how more sustainable systems can help improve that.

In Chapter 3, we presented the methodology for the learning systems that will control the entire solution, from data analysis to the node self-configuration, based on Machine Learning techniques. For that, after an overview of the field of Machine Learning and an introduction to all the necessary concepts, a study was performed to understand which were the best algorithms to implement on our solution. A training methodology was created and tested in several applications, in order to understand if there was an overall best technique or if for different scenarios, different approaches were needed. After that, a comparison between cloud and edge computing was performed to understand how this

can improve the system effectiveness, efficiency, latency and energy consumption, in terms of data analysis. The work presented in Chapter 3 resulted in the following publications:

- A. Glória and P. Sebastião, "Comparison of Edge Computing and Cloud Computing for Data Analysis in Sustainable Smart Nodes supported by Machine Learning", IEEE Access, 2021 (under review)
- A. Glória, M. I. Pires, J. Cardoso, J. Alves Coelho, and P. Sebastião "Drought Prediction for Landscape Sustainable Irrigation Using Random Forest," IEEE 10th International Conference on Intelligent Systems (IS20), 2020, pp. 10-15
- A. Glória, J. Cardoso, and P. Sebastião, "Improve Energy Efficiency of Irrigation Systems using Smartgrid and Random Forest," 2020 5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM), 2020, pp. 1-6
- A. Glória and P. Sebastião, "Temperature Distribution Analyses with Wireless Sensor Networks and Machine Learning" 2019 International Conference on Sensing and Instrumentation in IoT Era (ISSI), 2019, pp. 1-6

In Chapter 4 we applied our learning system methodology to create an autonomous communication configuration system based on Machine Learning, capable of self-configuring IoT smart nodes with the best communication protocol based on location and available protocols. For that, the chapter starts with an introduction to wireless communications in IoT systems, followed by a practical study on which is the best protocol to communicate between end devices, based on implementation environment, line of sight, distance and transmission power. Knowing how each protocol behaves in different scenarios, it was possible to create and test a methodology for the autonomous configuration system for point-to-point communications, based on a Machine Learning approach. Also a methodology for an autonomous configuration system for cloud communications was developed and tested. The work presented in Chapter 4 resulted in the following publications:

- A. Glória and P. Sebastião, "Autonomous Configuration of Communication Systems for IoT Smart Nodes supported by Machine Learning", in IEEE Access, vol. 9, pp. 75021-75034, 2021
- A. Glória, C. Dionísio, G. Simões and P. Sebastião, "LoRa Parameters Self Configuration for Low Power End Devices" 2019 22nd International Symposium on Wireless Personal Multimedia Communications (WPMC), 2019, pp. 1-6

In Chapter 5 the development and implementation of the sustainable modular IoT node is described, alongside with all the necessary hardware and software needed to create the node and implement the edge computing self-configuration. Several hardware approaches were tested, to understand which was the best computing core for the node, as well as the modularity, in order to assess if it has advantages over a single purpose approach. For the self-configuration, multiple Machine Learning models were developed and tested, to create a system capable of identifying the attached modules and necessary features. Also, the implementation of the smart nodes, its costs, power consumption and dashboard are presented.

In Chapter 6 the developed system was integrated and tested in two Smart Cities use cases, in order to assess the effectiveness and efficiency of our new methodology facing the traditional methods as well as the use of our modular approach facing a standard IoT system. For that, our system was implemented alongside the traditional method and the standard IoT solution and the results obtained from all the approaches were compared in terms of effectiveness, efficiency, energy consumption, installation and maintenance. The work presented in Chapter 6 resulted in the following publications:

- A. Glória, J. Cardoso and P. Sebastião, "Sustainable Irrigation System for Farming Supported by Machine Learning and Real-Time Sensor Data", in Sensors, vol. 21, no. 9, p. 3079, Apr. 2021
- J. Alves Coelho, A. Glória and P. Sebastião, "Precise Water Leak Detection using Machine Learning and Real-Time Sensor Data", in IoT, vol. 1, no. 2, pp. 474–493, Dec. 2020
- A. Glória, C. Dionisio, G. Simões, J. Cardoso and P. Sebastião, "Water Management for Sustainable Irrigation Systems using Internet of Things", in Sensors, vol. 20, no. 5, p. 1402, Mar. 2020

In Chapter 7, we conclude the thesis and discuss future directions of research.

All experiments presented in this thesis were validated using real designed hardware.

1.3. Other Scientific Contributions

During our research, we have supervised a total of seven master thesis not directly related to the topic of this thesis. These studies are related to the use of traditional IoT systems to create more sustainable tasks or processes. These contributions have resulted in the following publications:

- B. Dias, A. Glória and P. Sebastião, "Prediction of Link Quality for IoT Cloud Communications supported by Machine Learning", IEEE World AI IoT Congress, 2021
- F. Raimundo, A. Glória and P. Sebastião, "Prediction of Weather Forecast for Smart Agriculture supported by Machine Learning, in IEEE World AI IoT Congress, 2021
- J. Cardoso, A. Glória and P. Sebastião, "Improve Irrigation Timing Decision for Agriculture using Real Time Data and Machine Learning", 2020 International Conference on Data Analytics for Business and Industry: Way Towards a Sustainable Economy (ICDABI), 2020, pp. 1-5
- J. Cardoso, A. Glória, and P. Sebastião, "A Methodology for Sustainable Farming Irrigation using WSN, NB-IoT and Machine Learning," 2020 5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM), 2020, pp. 1-6
- A. Glória, C. Dionísio, G. Simões, P. Sebastião, and N. Souto, "WSN Application for Sustainable Water Management in Irrigation Systems," 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), 2019, pp. 833-836
- C. Dionísio, G. Simões, A. Glória, P. Sebastião, and N. Souto, "Distributed Sensing Solution for Home Efficiency Tracking," 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), 2019, pp. 825-828
- G. Simões, C. Dionísio, A. Glória, P. Sebastião, and N. Souto, "Smart System for Monitoring and Control of Swimming Pools," 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), 2019, pp. 829-832

Besides the publications, during our research over the last three years we were invited to share our research results as invited speakers in the following events:

- "Sustainable Irrigation System for Farming Supported by Machine Learning and Real-Time Sensor Data", in 2nd Smart Farm COLAB International Wednesday Meeting, Torres Vedras, Portugal, June 2021
- "Tackling Sustainability Problems using IoT: An Academic and Industry Overview", in Webinar on Emerging Trends in ICT for Entrepreneurship and Innovation, Beirut Arab University, Lebanon, December 2020

- "An Approach on Sustainability using Internet-of-Things and Machine Learning", in Workshop on Communications, Public Safety and Innovative Applications, ConfTele 2019, Lisbon, Portugal, June 2019

CHAPTER 2

State of Art

This chapter presents a state of the art knowledge of the sustainability challenges created with the Smart Cities proliferation by focusing on how this new reality is creating new research challenges and how it needs to be improved in order to be fully deployed on a worldwide level. It starts with a review on the proliferation of IoT and Smart Cities and the technologies that composed them, including sensing, communication and data analysis. Based on these concepts and technologies, the main research questions and challenges are detailed and analysed, including an identification of the most important challenges to focus on and how they can be solved. Also, some related work on sustainable solutions already developed are described, in order to create a basis for our research and development. Finally, a remarks section closes this chapter with a brief discussion and conclusion taken.

2.1. The Proliferation of IoT and Smart Cities

This century is defined by the increase in urbanization throughout the world, with more than 55% of the world's population living in urban areas and an expected increase to 68% by 2050. Another trend that has proliferated in the last decades was the digitalization of services, technologies and products, with an exponential growth leading the Information and Communication Technologies (ICT) to be one of the critical aspects that enable the daily life of millions [**10**].

These changes lead to the vast proliferation of IoT and Smart City projects, with tens of billions of sensors and actuators being deployed around the world's urban areas. As these tend to grow exponentially, residents can interact and experience cities in a new way, transforming communities and creating new opportunities on the economic, social, security and sustainability aspect of those cities [**10**].

With these rapid proliferation, and due to the characteristics of these projects and technologies, some challenges also appear that can put at risk the advantages and implementation of Smart Cities, being important to understand how they work and the best way to implement them.

2.1.1. Internet of Things

Intended to play a fundamental role in our daily lives, IoT is nowadays being more and more incorporated into every service or product, as it increases our quality of life [**11**]. Everywhere from Smart Homes, where users can control their thermostats or lights with a Smartphone, to mail shipping, where real time sensors can tell the condition of the package, IoT is improving the efficiency and overall satisfaction of process that until a few years back were done manually or not even considered [**12, 13**]. IoT research and implementation has been growing in the last years, connecting real world elements and adding intelligence and communications for smart process and autonomous decisions. IoT is enabling different types of beneficial applications and services that can sustain our day-by-day in ways that we never expected before [**14**]. For this IoT monitors and automates through the use of sensors and actuators networks and intelligent processing units, that share information between devices and allows them to work together to improve user experience.

Studies from J. A. Stankovic [**15**] shows that IoT research and implementation main topic focus on Transportation, Smart Houses, Smart Cities, Lifestyle, Retail, Agriculture, Smart Factories, Supply Chain, Emergency, User Interaction, Healthcare, Culture and Tourism. This, allied with the business layer of IoT, proves the array of possibilities that can reach new markets in upcoming years. But for these to become reality, there is a need to be able to connect objects, or "things", in an easy and reliable way that can adapt to any situation.

The literature shows a variety of ways in which IoT is composed. In 2015, A. Al-Fuqaha et al. [**16**] explained that IoT can be divided into six elements [**16**] that help us understand its true meaning and functionality. These are, identification, how the sensors and actuators are identified; sensing, the ability to gather data from the environment; communication, how devices are connected; computation, how data is analyzed; services and semantics, how the knowledge is collected and actions are created. More recent works divide IoT in just three or four components. M. Nasiri et al. [**2**], as shown in Figure 2.1, described that IoT is divided only in things-oriented (sensors), everything that creates a connection between a device in IoT; internet-oriented (middleware), everything connected to the network; and semantic-oriented (knowledge), everything linked with management, such as storage, search and data processing [**11, 17**].

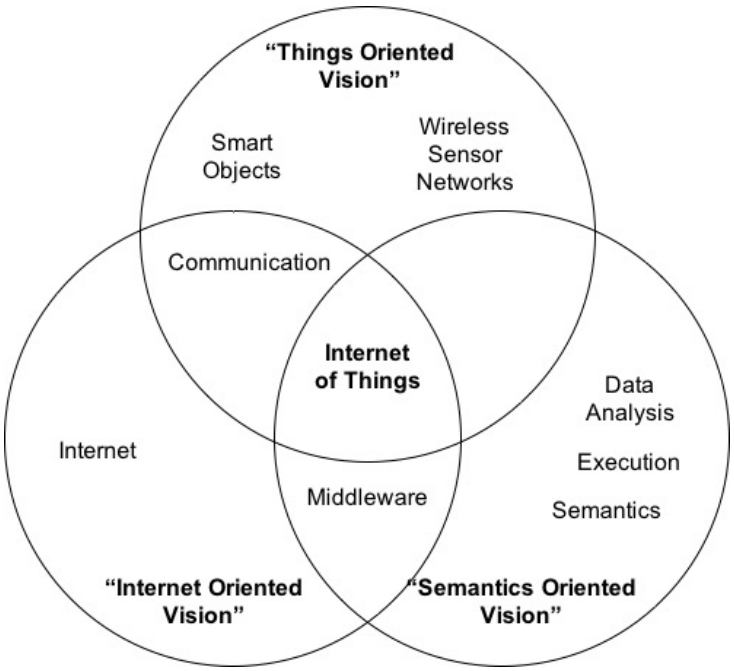


FIGURE 2.1. IoT Elements (stated by M. Nasiri et al. [**2**])

M. Albreem et al. [**9**] and R. Reddy et al. [**18**] state that IoT is divided in four main elements: Hardware, sensors, tags and others, that collect data, actuators and transceivers, that provide the ability to connect to the "things"; Communication, a way to transmit the data collected and actuators functions, from the "things" to the servers, either by Internet or other Wireless communication protocols; Middleware, for data storage and analyses, in order to collect knowledge from the "things"; and Visualization, a way to present data to the users, control the "things" and others via an application or dashboard.

IoT had major advances over the last years, with new compositions appearing as new architectures, hardware, software or functionalities are included. It is safe to say that IoT is a dynamic environment in constant change and evolution but that will always include four major components, as shown in Figure 2.2, that are interconnected to create an IoT system capable of having a positive impact on the site were it is implemented.

(1) Sensing: The ability to collect data from the devices;
(2) Actuation & Visualization: The ability to perform upon the environment or interact with the users;
(3) Communication: The ability to interconnect devices with other devices or the user;
(4) Data Analysis: The ability to gain knowledge from the collected data.

FIGURE 2.2. IoT Components

2.1.2. Smart Cities

For years, cities and companies have collected large amounts of data about cities, but those were often ruled out due to its small number of variables, low continuity in time and limited access [**6, 19**]. With the advances in IoT in the last few years, with current technologies and communication methods, this is no longer an issue, contributing to the boost of Smart Cities (SC). Nowadays "smart" is not only having data available but the process of closing the loop, with sensing, communicating, analyzing and actuating, as shown in Figure 2.3 [**4**]. Smart Cities tend to be more and more implemented due to urbanization of cities and the need for constant monitorization to not only prevent certain events, but also improve citizens' lives. For this, Smart Cities uses all the available resources to monitor conditions, gather data through sensors and manage infrastructures. Then uses all of the above to offer citizens better conditions [**20**].

In 2010 IBM stated [**21**] that SC are "the use of information and communication technology to sense, analyze and integrate the key information of core systems in running cities". Since then, several SC denominations appear in the literature: in 2011 K. Su et al [**22**] and A. Cocchia, in 2014 [**23**], stated that SC is more than a digital city, when combined with IoT, because SC intends to improve citizens lives through the development of new services that impact the economic, social and political progress as well as the protection of the environment. Mitchell, et al. defines SC as a city that works in a sustainable and smart way, integrating all its infrastructures and services into a unified process, using intelligent devices for monitoring and control, ensuring sustainability and efficiency

FIGURE 2.3. Infrastructure for Smart Cities [**4**]

[**5**]. Gartner, Inc., defines SC as "an urbanized area where multiple sectors cooperate to achieve sustainable outcomes through the analysis of contextual, real-time information shared among sector-specific information and operational technology systems." [**24**].

Besides all definitions, from various points of view, none was formally accepted, but all agree that Smart Cities final aim should be to make better use of public resources while increasing the quality of services and reducing waste, in both resources and monetary, in order to create better environments for the citizens. Smart City must be defined as the next big thing in the process of urbanization and digital development, with factors such as capital, civic engagement, government and environmental issues becoming the biggest drivers for the success of Smart Cities [**5**].

This emerging concept aims to disruptively enhance the efficiency, sustainability, and safety of urban communities, with integrated infrastructure and services, that allows monitoring and management using intelligent devices and systems [**7, 4**].

The core of Smart Cities relies on a network of data collection based on IoT [**25, 16**], with an accurate, distributed and real time sensing platforms and high performance communications structures, that are attached to the infrastructures of current cities. This allows municipalities to monitor and respond to changing conditions within the community in real-time, creating intelligent decisions and statistical correlations [**26, 7**].

According to [**5**] a Smart City can be divided into three core categories that together create valuable solutions. The first, and the most significant, is technology, both hardware

and software infrastructures that are the key feature in Smart City solutions, since the correct use of ICT can lead to the enhancement of life and work as well as sustainability within cities and their citizens. Second it is the Humans, the ones that identify the needs for a Smart City and those who developed said solutions. Those are the end users of Smart Cities and those who Smart Cities aim to help, so an effort to involve people to use Smart Cities solutions and take full advantage of them is critical. Lastly, there are the Institutions or Government, since most of the Smart Cities projects will need approval or will be supported by the city government, a set of rules and policies is fundamental to design and implement Smart Cities projects. Another look at this division done by Zanella et al. [**27**] divides the Smart Cities components into smart people, smart governance, smart mobility, smart environment and smart living. All of these are contributing for many city governments [**24**] to embrace the concept of Smart Cities in order to increase operational effectiveness and to satisfy the needs of their citizens and businesses.

Smart Cities are now more practical and productive in terms of implementation, but there are still many areas that need more research. As H. Habibzaleb described [**26**] almost none of the Smart Cities applications are static, they need to be integrated and complement the existing infrastructures in order to function properly as a system. Data acquisition and validation, system deployment and sustainability and privacy are some of the main areas that need to be investigated.

2.2. Enabling Technologies

A complete service in the Smart Cities context, or a normal IoT product, is composed of a set of modules that, when connected, create an end-to-end package. In order to collect information from the environment and be able to act upon the same, a process that includes gathering, transporting and analysing this information must take place.

The overall framework for intelligence in Smart Cities usually includes three levels: Smart City and IoT infrastructure, fog computing, and cloud computing, as can be seen in Figure 2.4. The lowest level, the IoT infrastructure, is where the sensors and devices perceive the environment, retrieve data and send them to the fog level, since this layer does not typically have a direct connection to the Cloud. In the fog computing level, the transceiver receives the raw data sent from the sensors and transmits them to the cloud computing level. Also in this level, some computation can be done, to reduce the amount of data sent, from noise, redundant or useless information. At the cloud level, complex and large-scale machine learning algorithms are employed to extract information from the

data collected and learned in order to provide knowledge and improvement to the Smart City infrastructures.

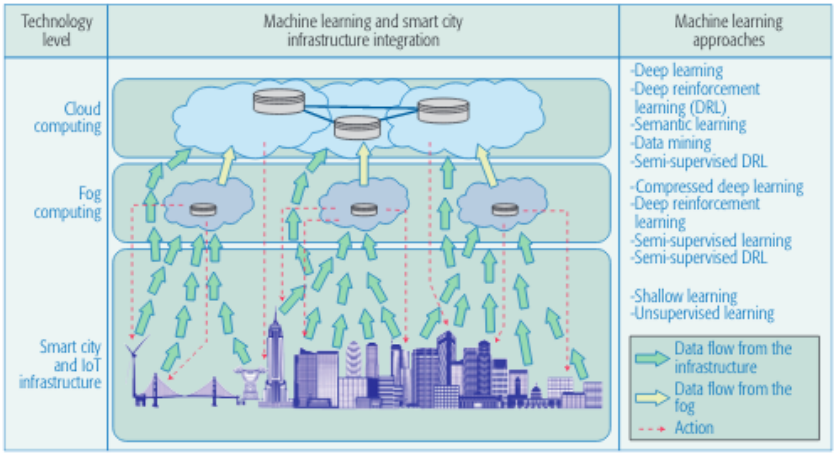


FIGURE 2.4. Levels of Intelligence in Smart Cities [**25**]

Since these are all different research areas, from Hardware, Electronics, Wireless Communications to Software and Data Science, it is common to divide the projects within these fields, with individual groups, and combine all the research at the end, in order to create the best possible service.

2.2.1. Sensing

Ever since hardware products exist, sensor networks can be found in order to retrieve critical information. But for decades these networks were closed or only locally accessible, which means that data was only available when someone collects them. With the rise of wireless communications, also the sensor networks become wireless being the perfect solution for distributed sensing solutions that interconnect several sensors and then use the existing communication network, such as Internet or cellular, to send data to the servers or users, thus creating the founding basis for IoT.

IoT depends on the ability to collect, send and process data, so the need for sensor networks is critical. These type of networks, as shown in Figure 2.5, are composed by a number of nodes that communicate in a multi-hop way, being these nodes any device equipped with sensors, such as smart-phones or cars or even amateur micro controllers, as Arduino or Raspberry Pi. These must be scalable, reliable and robust. Scalable due the high number of nodes that this network can have, reliable because in some cases these networks can send warnings in case something is wrong and robust because sensor nodes can be exposed to failures or bad environmental conditions [**11**].

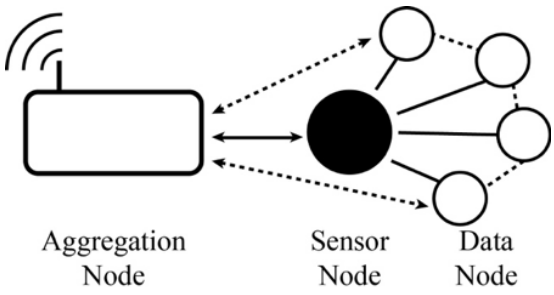


FIGURE 2.5. WSN Architecture

In a Smart Cities context, two categories of sensors can be defined [**26, 28**]. Hard sensing is a paradigm where sensors are tailored to precisely meet the application requirements, meaning that sensing station are placed in specific location to measure certain predefined parameters, with a specific architecture composed of sensors, processing, publishing and control, as shown in Figure 2.6, and that are owned by the administrator of the service. In a more decentralized basis, soft sensors are composed of non-dedicated sensing solutions, such as mobile phones or smart cars from the citizens or other mobile sensors, that ensure a more veracity and efficient sensing solution, making soft sensing an essential component for Smart Cities. Due to miniaturization of navigation systems, camera, accelerometer, gyroscope and microphones being the most used soft sensors, and as mobile devices are equipped with this type of sensors, by the end of 2018 soft sensors exceed half of the world population [**29**].

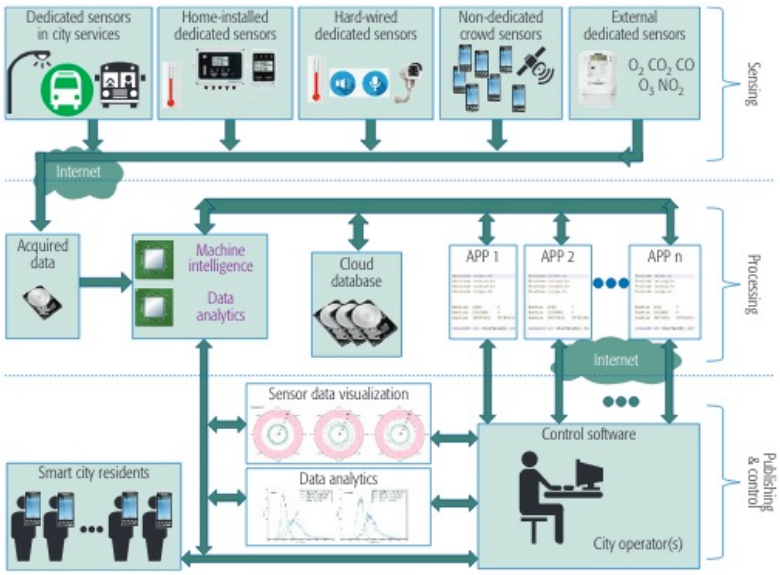


FIGURE 2.6. Components of a Smart City [**26**]

As shown in Figure 2.5, there are two types of nodes inside a WSN [**30**]:

- Edge Devices, low power, low resource devices containing sensors or actuators, with a single purpose of collecting data from the environment and reporting them to the gateway;
- Gateway Devices, responsible for connecting the edge devices with the Internet, aggregating the data sent from them and routing it to the needed service, and vice-versa.

Since WSN are a big part of the Smart Cities context and with the amount of devices deployed into Smart Cities projects, expected to be more than 100 billions by 2030 [**9**], efforts to make them more "green", mainly in terms of power and storage, is an effort with a tremendous importance.

2.2.2. Communication

IoT gains its strength from the connection with the network and with the internet expanding further every day, with a potential 100 billions connected devices in the future, communications are a big part of every project. New protocols are appearing and old ones are adapting to this new reality. IoT communication can be divided in three major components [**18**]:

(1) Device to Device (D2D), used within the sensor network in order to exchange data between nodes, mainly by Wireless Communication protocols such as Bluetooth, ZigBee, Long Range (LoRa), ESPNow, among many others. Also, some Wired Communication protocols are still in use, as I2C, RS232/485, SPI or CAN-bus;
(2) Device to Server (D2S), used to send the gathered data from the network to the servers, based on Internet connected services such as Wireless Fidelity (Wi-Fi), with Message Queuing Telemetry Transport (MQTT), REST, CoAP and others, but also Ethernet or cellular communications, such as NarrowBand IoT (NB-IoT) or 5G;
(3) Server to Server (S2S), used to exchange data from storage to processing, for example.

To achieve a network capable of adapting in almost every scenario a conjunction of all 3 types is necessary. The main focus is always in D2D communications, since most projects have a central node that is connected to the Internet using MQTT as D2S, and all of the end nodes, that can be in long range of that node, use wireless protocols to send data to it. One of the biggest problems is the lack of coverage or reliability of communication protocols in certain environments, mainly when comparing indoor and outdoor projects

or city versus rural environments. In the literature, regarding WSN projects, it is possible to see that throughout the years the preferred standards, mainly in communication, have changed. From 2014 up to 2016, ZigBee was the chosen communication protocol for almost every WSN project, but as stated in [**31, 32**], ZigBee cannot keep up with current expectations in cost, range and power, being Low-Power Wide-Area Network (LPWAN) protocols such as LoRa or SigFox the main successors in more current projects. Also cellular protocols, such as NB-IoT or 5G, are being applied in solutions that eliminate the central node, as they can send data directly to servers. WiFi and Bluetooth Low Energy (BLE), are some of the technologies that are adapting to this new reality and are still important features in such projects.

2.2.3. Data Analyses

Computation is one of the major components of every IoT and Smart Cities project, as it is how everything is controlled and where decisions are made. The key role of computation is to analyse all the information that is collected by the system and create knowledge from them in order to understand changes in the environment.

With the recent evolution of IoT and big data analytics, Smart Cities are more and more a reality, as more data and intelligence are easily achieved. Due to the recently massive application of WSN projects, that attracts attention due to its amount of data and need of self-adaption, resilience and cost-effectiveness, features that can be achieved with a proper analysis of that data, more knowledge and intelligence can be applied to those projects. Monitoring and control systems achieve their prediction by manipulating captured data and feeding them into machine learning algorithms.

As described, one of the major problems with the proliferation of IoT and Smart Cities solutions is the amount of data that is being generated every minute, becoming impossible to analyze the data manually and therefore necessary to have technologies with the ability to interpret and process this data without human intervention, and this is how Machine Learning arises.

Machine Learning is a perfect fit in a Smart Cities context due to the amount of data that is created and the inability for humans to process all of it in real time. Since a Smart City is a dynamic environment where a continuous learning mechanism is needed for applications that are not fixed, instead evolves over time, and due to the amount of noise and uncertainty that comes from mass data environments, the necessity to adapt,

in real time, in order to learn and improve the analysis by itself from previous experiences is a requirement [**25**].

Machine Learning gives the system the capability of learning from previous experiences, being able to predict future events and create decisions based solely on data, without human intervention. As more and more data or experience is available to machine learning a more efficient and better performance decision can be achieved [**33**].

As such, with the rise of Smart Cities, the data gathered by the deployed sensors can be used, alongside advanced technologies and data analytics to improve the efficiency of city services and residents' quality of life [**34**]. With these technological advances, the combination of smart devices with data analytics provides Smart Cities the capacity to interconnect all of its services, creating smarter homes, infrastructures and services, improving their efficiency, privacy, security and quality of live/work [**35**]. The application of data analysis can be found in transportation, power and water distribution, traffic management, healthcare, environment and even agriculture.

2.2.4. Actuation & Visualization

With the ability of collecting, transmitting and analysing data, IoT not only is capable of understanding what is happening but can also provide a platform for interaction with the environment.

As IoT and Smart Cities solutions are widely growing, the need to provide an easy way to check all the collected data, for both information systems and final users, was necessary. With the interconnectivity of IoT systems and cloud services, with just a simple smartphone it is possible to connect to an entire network of sensors of a Smart City and visualize all the data and even make modifications and configurations in real time. With web or mobile applications, and cloud services such as Amazon Web Services, Microsoft Azure or IBM Watson providing the middleware between the network and the visualization platform, nowadays every IoT system is not completed without an application.

These types of visualization platforms provide access to a set of features associated with the system, including the visualization of real time and historical data, receive alerts and notifications about the system conditions and even perform remote modifications to the system. Besides visualization and system configuration, it also allows for the control of system actuators.

With the uprising of more complex IoT projects, a new type of networks start to appear, making use of an extra type of nodes. The Wireless Sensor and Actor Network (WSAN) is composed by an addition of an Actor Node, a more resourceful and rich node, equipped with better processing capabilities and higher transmission power [**36**]. Designed to act upon the environment, they are usually coupled with motors, lights, or any other actuator that can be controlled by the node.

With the visualization platforms it is possible to turn ON or OFF these actuators in a remote way, but with the integration of data analysis, and to create autonomous system capable of adjusting its configuration to reach a goal, it is possible to control the actuators based on the sensor data that is collected, for example turning ON a Heating, Ventilation and Air Conditioning (HVAC) system when the temperature rises above a threshold value.

With the introduction of visualization platforms and actuators into IoT projects Smart Cities are able to be a true autonomous system capable of being controlled remotely and to adapt themselves to face any situation.

2.3. Main Issues

Smart cities are more and more part of our world, but for the complete proliferation of these technologies and in order to make them more appealing to the average users, some issues still need to be solved, since Smart Cities applications are not static, they need to be integrated with other applications. New IoT solutions appear every day, but most of the research done in these areas are not on how to create new solutions or projects but how to approach the existing challenges. With the rapid advance in sensing, communication, storage, embedded systems and processing power on IoT systems, and the deployment of these solutions to millions of peoples, new challenges emerge that can potentially put in risk the benefits of Smart Cities.

The literature [**18, 9**] shows that the main issues facing IoT and Smart Cities are as follow:

- Data Provenance & Trust [**37, 38, 39, 40**]
- Lack of heterogeneous protocols & standards [**41, 42**]
- Learning Systems
- Privacy & Security [**43, 44**]
- Power Devices & Battery life [**45, 46, 47**]
- Storage

- Amount of devices/communication

All these problems, as will be described, create concerns about the environment and health, opening research areas in the need to create technology in green renewable ways.

2.3.1. Data Provenance & Trust

With the amount of new solutions that appear each day, with the proliferation of inexpensive sensors, and due to the lack of structures and tags, by 2020 more than half of the stored data will be non-useful information. So, to create better data quality, acquisition must be complemented with structuring and tagging [**26**].

New data acquisition techniques are needed to ensure high-quality data, thereby increasing the veracity of soft sensor data. Valuation of data and proportional incentives (to convince users to offer their non-dedicated resources for use) also need to be investigated. Guaranteeing the quality of the collected data is a big issue on IoT projects. With scarcity and irregularity of data, being some of the issues to take into account when discussing about Smart Cities data, now the big challenge becomes the ability to process huge amount of data, that are filled with useless or redundant sources, and extract real useful information [**6**]. The heterogeneity of the current devices and sensing platform creates a need to think in the veracity and trustworthiness of sensed data, with a need to detect fraud and misinformation, malfunctioning sensors and misuse of devices, that can affect the data analysis as H. Habibzadeh et al. thoroughly described in [**26**].

2.3.2. Lack of heterogeneous protocols & standard

With the number of IoT solutions and devices increasing exponentially in the most recent years, also the number of proprietary protocols and cloud services grows, creating an interoperability between devices and impossibility to share data between services [**48**]. This "vertically services", a closed and/or proprietary systems dedicated for a particular task, which can be combined easily with third party components, are a big challenge for Smart Cities, as it when an investment is made in one of these services the city is locked into that particular vendor's system [**48**]. For Smart Cities to work, open data needs to be a reality, global data collection needs to include data from every government, industry and general public [**48**]. To create new solutions, datasets need to be available and right now, due to the lack of standardization and open data, there is a shortage of real-world datasets to train new models and test new solutions, in order to confirm the results obtained with simulated data [**25**].

2.3.3. Learning Systems

With the amount of real time data that IoT and Smart Cities solutions generate, preparing and processing these are critical, not only because of the lack of protocols, as said before, and as different sensors give data in different formats, a quick and efficient analysis is critical [**18**]. Also due to integration, easy scalability and security violation in data, this is a major challenge.

With many of the Smart City applications requiring real-time analytics, new frameworks that support this combined with fast streaming data analytics are required [**25**]. But also some lightweight analyses, done directly on the end devices, is necessary and can help improve security and privacy, since data is not transferred to the cloud [**25**].

Another approach needed in the learning systems is how to prevent false data to be used to attack the Smart Cities. Validity and trustworthiness of data driven machine learning approaches can easily be put down with the injection of false data [**25**], either by replacing the sensors with false ones or by jamming the communications and replacing the data. Learning systems need to understand if the reading received from the sensors are real or if they are somehow tempered with.

2.3.4. Privacy & Security

If there is a number one challenge in Smart Cities it has to be privacy and security of the data collected and how they be directly linked to the citizens, since the larger amount of data in a Smart City environment comes from individuals who may not like to see their data publicly available [**25**].

IoT systems present vulnerabilities due to its network architecture, where physical devices collect information and transmit them via wireless connections to be processed and stored. This provide a perfect scenario for man-in-the-middle and denial-of-service attacks, where data is retrieved, changed or misplaced to affect the solution, impacting the security and privacy of Smart Cities [**49**]. Besides the wireless communication, that are prone to jamming and/or other attacks that replace information and send false data, also physical tampering is an issue, since the physical devices are spread outdoors and can be easily manipulated to create false data.

The battery powered devices usually used in the IoT and Smart Cities context are typically inexpensive and limited in terms of resources, not allowing the deployment with advanced security features, and since Smart Cities use a dynamic networks of devices, a centralized security network is impossible to implement [**50**].

As the data flow inside a Smart City has a high level of interaction, between people, devices and sensors, all of these components must be able to protect their data and assess and respond to external threats, using authentication, access control, confidentiality, trust and secure middleware. This way, not only is possible to create a secure information system but also to provide privacy, confidentiality, integrity and availability to Smart Cities and their users [**49**].

The different types and growing number of devices, applications, methods of communication, types of data and others, implies a huge amount of challenges associated with security within IoT projects [**30**].

2.3.5. Power Devices & Battery life

With the rise of devices and end nodes being deployed into the growing network of Smart Cities, one of the main concerns relies on how to power all these devices in a green way. Energy consumption need to achieve a green reliability and a world implementation, so IoT should be designed to be energy efficient and reduce greenhouse effects and carbon dioxide emissions of sensors, devices, applications and services, with a lifecycle as described in Figure 2.7 [**9, 51**].

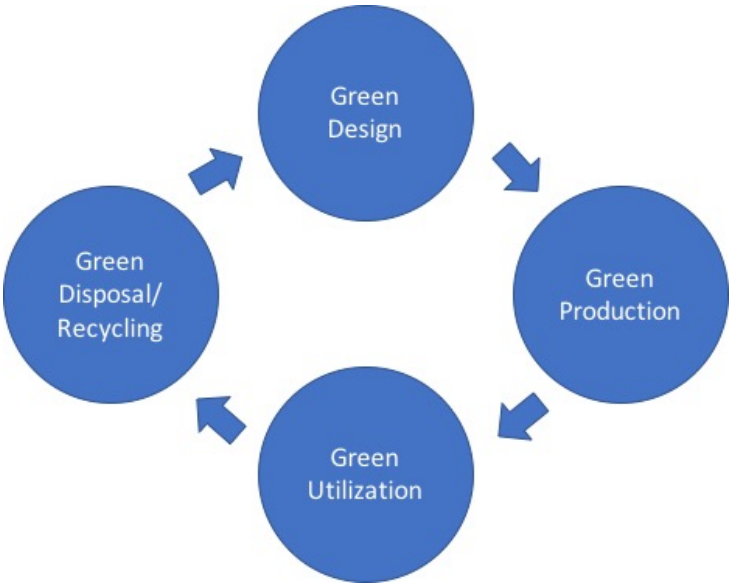


FIGURE 2.7. Life Cycle of a Green IoT Device [**9**]

Changes on how to transmit and process data also are being implemented in order to reduce the power consumption of end devices, since these are the ones that exist in higher numbers and whose batteries need to create a concern. As described by X. Fafoutis et al. [**52**] any data that is wrongly gathered, transferred, stored or processed is a potential waste of energy.

An IoT device must have the ability to power themselves over extended years without the need to changes or recharge the batteries [**53**] and since battery life is limited additional intelligence must be included to manage energy use [**4**], otherwise they cannot be considered as a green device. However, transmitting data is one of the more power hungry operations done by these devices, mainly if done over large distances, with a need for continuous power source or frequent charging [**54**], thus the ability to maintain power efficiency while transmitting data over long distances will be an important factor in IoT devices [**55**]. This also allows to reduce power by sending data to be processed in the cloud, instead of in-device analysis, which reduces the processing power done by the devices reducing the power consumption.

Other solutions have been presented from energy-efficient sensing [**56**], duty-cycling low power networks [**57**], energy-efficient security [**58**] and low power operating systems [**59**].

A solution for this problem can be achieved with different techniques [**9**] that can be taken into consideration when developing new systems:

- Harvest the energy from the activity, avoiding additional external power sources;
- Put the nodes into idle or sleep mode when not in use;
- Use renewable energy for charging;
- Optimize the software and hardware to be energy-efficient;
- Reduce data size, and when possible retrieval circles, to reduce the amount of data stored.

2.3.6. Storage

In an IoT architecture, previously collected data is used to create future decisions, based on machine learning algorithms, so storing all the collected data is essential. But this comes with the problems that a simple IoT solution collects tons of data. With the global spread of these solutions, storage needs to be adapted to this new reality. Since keeping track of all sensor data for future use is imperative, addressing storage challenges while data is growing is a must [**26**]. With each Smart City application needing huge storage capabilities, although storage prices are decreasing, keeping all these data in physical locations is not a good policy [**60**], as the energy cost of data transfer and storage is about 7 kWh per terabyte per year, that corresponds to the emission of 20 kilograms of CO_2.

Data compression into single equations can be a solution to store more data using less space, since most of these data is produced periodically over time, making it possible to create a regression that can represent all the data collected as well as predict future data [**60**] Another solution is to have end nodes, with some computational power, with methods such as duty-cycling or verification that can be coupled with optimization models, in order to do some filtering to detect duplicated, irrelevant or redundant information before it arrives at the back-end [**26**].

2.3.7. Amount of devices/communications

IoT and Smart Cities need devices to work as intended, with new solutions appearing every day and replacing the previous solutions. This growing number of devices, that is expected to reach 100 billion connected by 2030 [**9**], will lead to an even growing number of malfunctions and discarded devices. Due to the lower costs of these devices and solutions, many of them will consist of impulse buys that will end up being discarded, due to the inability of installation or usage, connectivity or battery problems, or will be misused, with over communication, power consumption and creation of unusable data.

This rise in the number of devices not only will increase drastically the amount of data exchanges, with 10000 times more in 2030 than in 2010, creating more congestion on the communication networks, but will also take a big impact on the carbon emissions into the environment [**9**]. The fast scalability of Smart Cities and its devices will have some critical issues in terms of environmental impact, with an exceptional amount of carbon emissions into the environment, with an expected 345 million tons of CO_2 already created until 2020, just for the communication and data exchange [**9, 61, 62**].

2.4. Environmental Impact & Challenges

In order to be deployed in a worldwide fashion, Smart Cities need to decrease this environmental and health issue, as cities nowadays are faced with rules and regulations to reduce CO_2 emission or fines if they do not comply.

As described in the previous issues, with the rapid proliferation of Smart Cities, it is expected that the rise of short-term use devices and lack of compatibility between manufactures will create a massive amount of device disposals; and that the massive use of these devices, as well as the need to store the massive amount of data, that increase the need for communications, infrastructure and maintenance, will generate tons of CO_2 being release into the atmosphere every minute.

In 2015, all the United Nations Member States approved the 2030 Agenda for Sustainable Development, providing "a shared blueprint for peace and prosperity for people and the planet, now and into the future" [**63**]. To achieve this they created 17 Sustainable Development Goals (SDGs), each focusing on an urgent call for action, from poverty, education, economic growth and the environment, with a set of strategies to improve or mitigate the problems associated with each of these.

And the sustainability of cities and communities is focused on the 11^{th} SDG. With the goal of "Making cities and human settlements inclusive, safe, resilient and sustainable", through the reduction of disaster risks, sustainable transport and cities [**64**]. To achieve these goals, this directive supports the use of technology and smart devices, in order to create autonomous mechanisms capable of improving the decision processes and reduce, or at least minimize, the situations where unsustainable events might occur.

But technology can not be the answer to sustainability, if that same technology contributes to the creation of unsustainable actions, for example, more CO_2 emissions. As such the challenge for new products and the deployment of these services into Smart Cities is that, not only they need to innovate and impact the processes inside cities, but they also must do it with the ambition or with regulations to achieve a reduction is terms of costs, carbon emissions, water consumption, use of plastics, among many other, facing the previous systems.

CHAPTER 3

Learning Systems

This chapter presents the research and developed solutions for the learning systems supported by Machine Learning. It starts with an overview on the research, containing the related work found in the literature, followed by an explanation of some of the most important definitions, including all the Machine Learning techniques and algorithms used as well as an introduction to cloud and edge computing. With all the definitions introduced, this chapter continues with the research done to find the best algorithm for each of the techniques presented, detailing the methodology followed, the scenarios dataset and tests performed and finally the obtained results and remarks. The last section of this chapter presents a study of the feasibility of porting a cloud computing model to an edge computing model, in order to assess not only if it is possible to run a full size model on a microcontroller, but also how it can be done, which modifications are needed, and how it affects the quality of the model. Once again, the methodology, test scenarios and results are presented and detailed. Finally, a remarks section closes this chapter with a brief discussion and conclusion of the topics and results given in this chapter.

3.1. Overview

Data processing requires large amounts of data, something that WSN and Smart Cities applications can provide, making the conjunction of those areas a very promising field for research. With the inability of humans to process all of this data in real time, Machine Learning can help improve, adapt and learn from the collected data. As Smart Cities are a dynamic environment where a continuous learning mechanism is needed since applications evolve over time, instead of being fixed, the analysis of the amount of noise and uncertainty that comes from mass data environments can be heavily supported by Machine Learning.

Sustainability can be largely improved by artificial intelligence algorithms, including prediction and deep learning algorithms, as it provides a way to analyze the gathered data in order to modify the systems behaviors to be more efficient and reliable. For example, see if the amount of water used by the irrigation system is enough to maintain the garden or crops healthy and if so apply the same amount to every system with similar conditions,

or understanding that the communication protocol used has a high packet loss rate in the current conditions and change the used protocol for that node. With this analysis, it is possible to give the client a system that can really adapt itself to every condition or specification in real time.

Machine Learning is the field focused on building applications that learn from data and improve their accuracy over time without being programmed to do so. 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 [**65, 66**].

Machine Learning and IoT are two fields that are more and more dependent on each other. With several studies conducted using Machine Learning techniques, it can be applied to anything from traffic light control, to smart farms, health care or industrial IoT. As our research is focused on learning systems for IoT, sustainability and wireless communications, and since the main goal of this thesis is to develop a system that is completely supported by Machine Learning algorithms, those were the focused areas when reviewing the literature to understand which were the best techniques, models and methodologies.

3.1.1. Machine Learning Techniques

Machine Learning techniques can be divided into three major categories: supervised, unsupervised and reinforced; each of them designed to tackle different situations and types of data.

For supervised learning, the goal is to train the model using a known output, so a dataset containing the the input and target values is used, being these values used to find a mapping function that can correlate the input and output values [**67**].

Contrarily, unsupervised training only knows the input data, being these not labeled or classified. The goal of the model is to find patterns, distributions of categories among these data, usually in terms of clustering [**67**].

Finally, in reinforced learning the goal is to train the model based on rewards facing the output. For each interaction, the model is rewarded and retrained based on that reward, allowing the model to learn based on the best interactions [**67**].

Based on the proposed methodology for the developed system, our focus will be in supervised learning, since our models will also work based on previous known data. Inside supervised learning, both classifications and regressions will be used.

3.1.1.1. *Classification.* In regards to classification methods, this one is known for its goal of approximating a mapping function from the inputs given by the dataset, in order to identify the output values. In (3.1) is shown the followed function by this method, where f is the mapping function, x the input value and y the output value [**68**].

$$\mathrm{y(f : x \rightarrow y)} \tag{3.1}$$

To evaluate the model performance, for classification, the Accuracy metric will be used, as it is the most common metric for classification. It measures the fraction of predictions the model got right, using (3.2).

$$Accuracy\ [\%] = \frac{Number\ of\ correct\ predictions}{Total\ predictions} \tag{3.2}$$

3.1.1.2. *Regression.* Regression analysis is used as a technique for prediction, by searching the relationship between a dependent (*target*) and independent variable (s). As the training data is independently selected from the original dataset, the mean-squared error for a predictor variable X for the class y can be estimated by (3.3) [**69**].

$$E_{X,y}(y - X)^2 \tag{3.3}$$

To evaluate the model performance, for regression, 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 (3.4) [**70**], where P_{rx} is the real value, $\hat{P}_{rx}$ is the estimated value, and N the number of samples.

$$MAE = \frac{1}{N}.\sum_{i=1}^{N} |P_{rx_i} - \hat{P}_{rx_i}| \tag{3.4}$$

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

3.1.2. Machine Learning Algorithms

In our literature review, and following the work done by [**71**], that did a comprehensive review of Machine Learning models for Smart Cities and sustainability, some techniques were highlighted as the most used, and the ones with best results, in the supervised learning area, for both classification and regression. Those included Linear Regression, Decision Trees, Random Forest, Neural Networks and Support Vector Machines.

3.1.2.1. *Linear Regression.* Linear Regression (LR) are the simplest methods to predict data, being adopted by Machine Learning as they came from the statistical world. It performs the task to predict a dependent variable value (y) based on a given independent variable (x). With this, it creates a simple linear relationship between the input and output, that can be presented in an equation as $y = A + B \times x$, also called a plane or hyper-plane, as shown in Figure 3.1, being A and B the coefficients that the regression will define to characterize the output based on the input [**66**]. Besides being marked as inefficient and inaccurate by researchers [**66**], when compared to other Machine Learning techniques, is still commonly used for scoring modelling, since it gives a logistics distribution of the data.

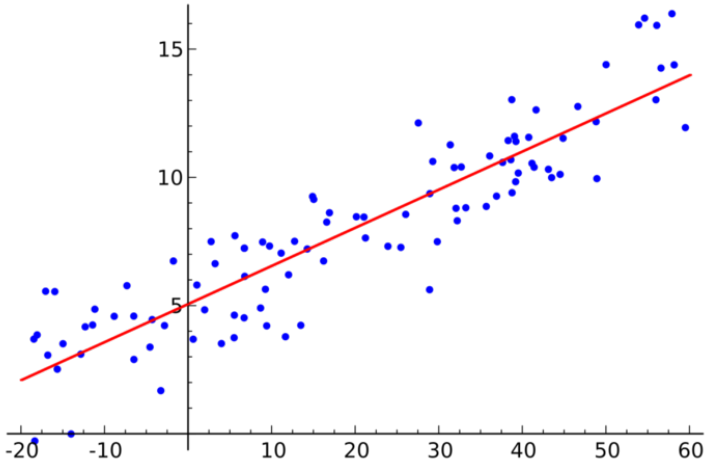


FIGURE 3.1. Linear Regression Working Logic

3.1.2.2. *Decision Tree.* Decision Tree (DT) are tree-based methods in which each path begins in a root node and multiple divisions are made, as can be seen in Figure 3.2, through a hierarchical partition of training data, taking into account the dataset. This creates sub-trees based on a certain features used to split the data, representing a sequence of data divisions, with this split being done iteratively until it reach a leaf node with an outcome, containing the number of records that can be used to classify the data [**66**]. These methods can be applied for classification and regression. The final goal of this method is to reach a model that can predict the search value for that specific scenario by learning simple decision rules [**50**].

3.1.2.3. *Random Forest.* Random Forest (RF) is a decision tree method, developed for classification and regression [**69**] and is composed by a large number of trees, each voting for the final outcome, being the final result determined by a majority vote from all

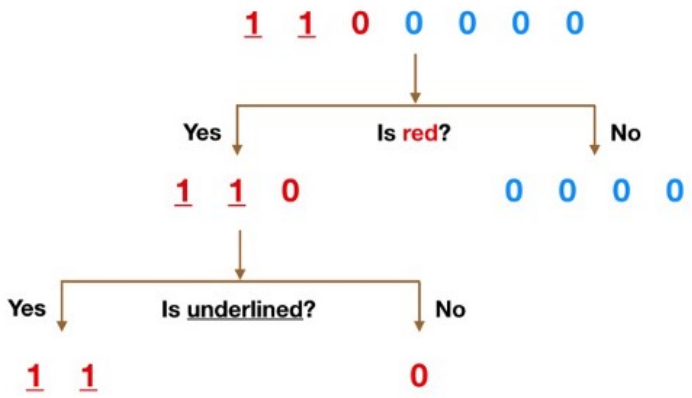


FIGURE 3.2. Decision Tree Working Logic

decision trees [**72**], as shown in Figure 3.3. With a great performance in predictive tasks, it is ideal for analyzing large numbers of parameters [**73**], even with small datasets, being highly applicable for classification problems. Random Forest incorporates the process of aggregation, bagging and Decision Trees, with a selection of a subset of features from each node of the tree, avoiding the correlation in the bootstrapped set [**66**]. The generated forest can also get great performance when new data is added [**74**].

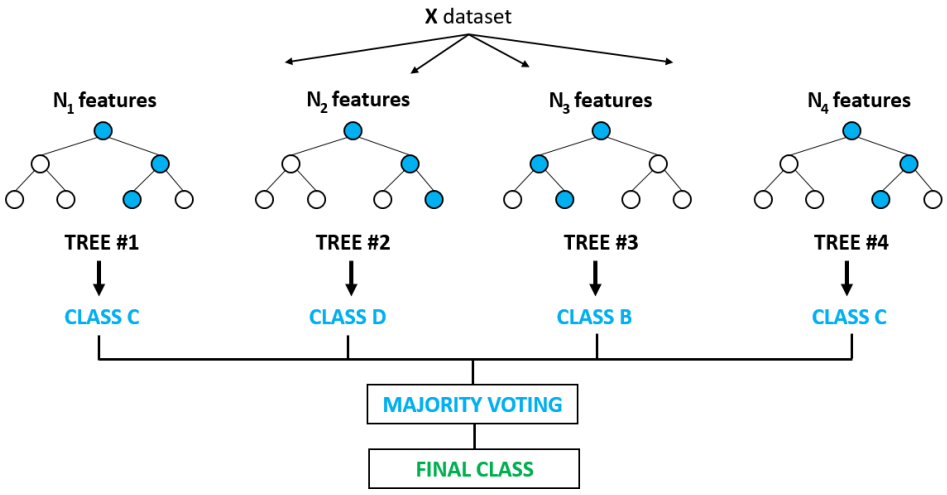


FIGURE 3.3. Random Forest Working Logic

3.1.2.4. ***Neural Networks.*** Neural Networks (NN) algorithms are defined as computational models of the neural system composed of several neurons connected one to the other by synapses, in the same way as the human nervous system. Each of the neurons analyzes parts of the input and sends the information to the next layer and neurons continuously, until it is able to reach a valid output [**66**]. This process continues until a final output is

found. It is ideally used in nonlinear and complex problems which require large computational power and has some disadvantages when working with IoT systems due to low complex and low power devices [**50**]. In this case, Multilayer Perceptron (MLP), a variation of the neural network algorithm which consists of multiple neurons organized into layers [**75**], was used. These MLP networks are characterized by being general-purpose, flexible and non-linear. Their complexity can be changed according to their application by varying the number of layers and units of each layer, as can be seen in Figure 3.4.

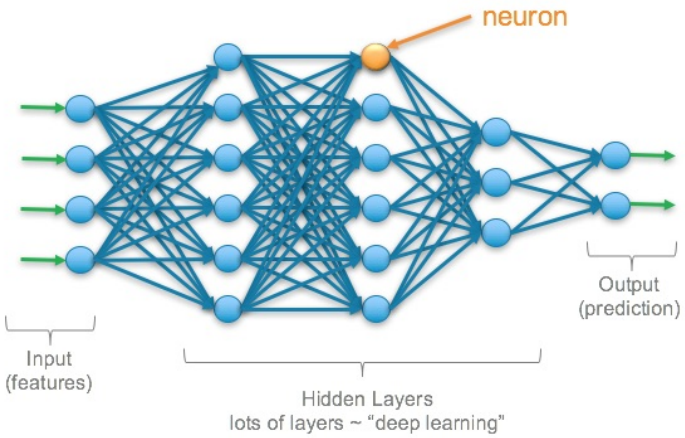


FIGURE 3.4. MLP Working Logic

3.1.2.5. ***Support Vector Machines.*** Support Vector Machines (SVM) use a hyperplane to create a decision boundary, as seen in Figure 3.5, in order to separate different classes of objects. Used for classification and regression, it uses complex mathematical functions to create this hyperplane and be able to assign the members of each class [**33**]. To discover the position of the hyperplane it uses a small subset of vectors from the training data, the support vectors, that define the edge of the class [**76**]. Mostly used for classification, since it can divide a dataset into classes, it lacks some probability estimate, mainly when large or more complex datasets are used [**33**].

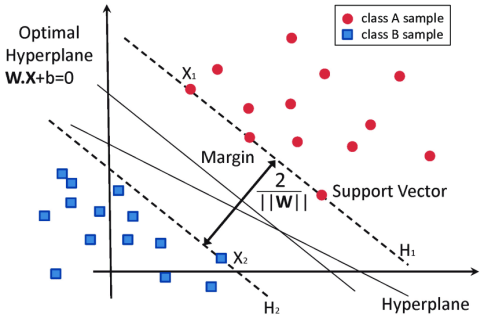


FIGURE 3.5. SVM Working Logic

3.1.3. Computing Techniques

With the increase of data providing from IoT devices and with the need to analyse and store all this data, computing emerges as one of the major roles in IoT services [**16, 2, 9**]. Computation represents the brain and computational ability of IoT, using hardware processing (e.g, microcontrollers, microprocessors, system on chips (SoCs)) and software or cloud applications to perform tasks [**16**].

As shown in Figure 3.6, there are three essential layers in an IoT project regarding computation [**77**]:

- Layer 1 – Sensors: formed by the IoT devices and its users, that are responsible for gathering information and performing operations upon the environment;
- Layer 2 – Edge Nodes: formed by more powerful nodes, in terms of computational power and features, that are responsible for maintaining the nodes networks, message exchange, storing data, data processing and some calculations;
- Layer 3 – Cloud Services: formed by cloud nodes that have higher computational requirements, such as Machine Learning, Business Intelligence, Big Data analytics, and visualization.

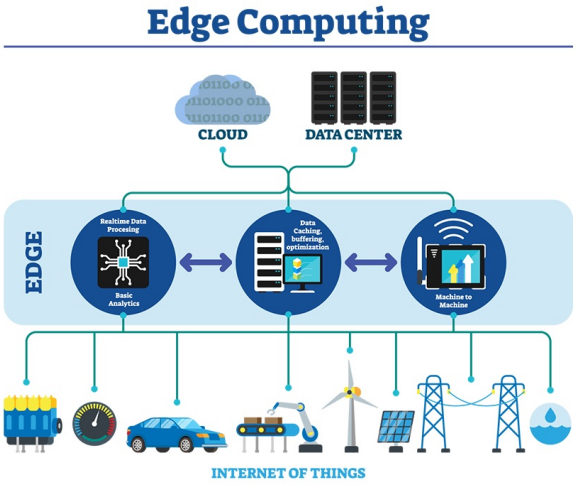


FIGURE 3.6. Computation Layers in IoT

With the proliferation of IoT and Smart Cities, the Cloud Computing techniques used in past years start to prove to have some limitations [**78**], so new techniques have emerged, such as Edge Computing, to improve cloud services.

3.1.3.1. ***Cloud Computing.*** With the increase of IT solutions over the last decades, Cloud Computing (CC) services provide easy, high performance computation with a low investment on servers, since cloud computing works on remote central cloud base servers.

As long as a device has an internet connection, it is able to send requests to the cloud and take advantage of all the available services, such as computing or storage, allowing also the connection between devices and services in different networks [**79**].

3.1.3.2. *Edge Computing.* With the proliferation of IoT devices, the need to reduce latency in data analysis, in typical remote central cloud base computation services, allow the adaptation of end devices to perform some of this computation, thus creating the Edge Computing (EC) [**80**]. EC solves some challenges of managing systems on a central cloud, such as response time, security and quality of service, by executing tasks closer to the IoT devices, with management, storage, data analysis and decision making being done directly on multiple edge nodes inside the network [**77**]. These interconnected devices and EC techniques prevent overload of computer processes, as well as, obstructions in the flow of data and the services that are sent or requested to the cloud [**78**].

3.2. Comparison of Machine Learning Algorithms for Data Analysis

In the literature it is possible to find several studies comparing Machine Learning algorithms, but none focusing on multiple outputs or scenarios and each one using a different methodology, making it almost impossible to compare among them which is the best algorithm overall.

Despite those studies, that individually provide good research when focusing on a particular field or output, our goal is to analyse the same algorithms under the same methodology, for multiple scenarios, in order to understand which is the model that has an overall advantage and that can be used in a modular and adaptable solution.

3.2.1. Methodology

To evaluate which are the best Machine Learning algorithms that fit our goal of evaluating data retrieved by sensor networks in order to predict outcomes in real time, each of the models was tested using following methodology:

(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 best results and which need to be improved to achieve them. The scikit-learn, an open source Machine Learning library developed for Python implementation [**81**], was the selected framework for the development of the Machine Learning models in the Anaconda environment.

(2) The obtained model for each algorithm was then submitted into an hyperparametrization tuning, that compares the model performance using different model configurations 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 [**82**];

(3) To guarantee that the model is stable, after finding the best configuration and training the model, a Stratified K-Fold 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 fold will produce a result, that is average at the end, allowing for a reduced error margin and variation, as more data is used to fit the model;

This methodology will, not only, be used to evaluate the algorithms presented in Section 3.1.2, but is also the methodology that will be followed for every learning system implementation used in this thesis.

3.2.2. Test Scenarios

The learning systems based on Machine Learning approaches are the basis for the entire solution presented in this thesis, not only in terms of self-configuration and autonomous maintenance of the system hardware, but also to help achieve the purpose in which they are installed in a real world scenario, analysing data in real time and deciding how to improve the efficiency of those tasks, mainly in a sustainable way.

As this solution can be implemented in multiple scenarios, each one using different data and needing different outcomes, the option by one single solution for the learning system could imply that in some scenarios it will have great results, but it can underperform in many others. As such, and to have a better understanding of which models work better in as many situations as possible or if a different approach for each scenario is needed, the evaluation of the Machine Learning models was performed for multiple specifications and different datasets and goals, in both regressions and classification problems.

The main problem with this particular methodology, is the need to have large datasets to train and test the learning systems. Although datasets are available online, it is hard

to find those who target sustainability or have the required parameters needed, mainly in terms of communication schemes.

To overcome that difficulty and train the models with data as close as possible to the one that will be gathered by our hardware, when open-sourced data or previously gathered datasets were not available, the datasets were chosen between the ones available inside the IoT research group led by Prof. Pedro Sebastião, in which I co-supervised a set of master students, as they use a similar architecture, in terms of hardware, sensors and communication, guaranteeing that the data can be reliable and that it will fit our methodology and framework.

The chosen scenarios can be found in Table 3.1 and include an HVAC operation dataset, to understand if the system is ON or OFF based on the temperature in multiple points; a water flow dataset, to detect leaks in the pipelines; an agricultural field conditions dataset, to adjust the irrigation hour based on the soil humidity conditions; a weather dataset, to forecast future conditions; a smart grid dataset, to predict the energy usage inside a house; and a wireless communication data transfer dataset, to predict the signal strength. Each of these are described in Appendix A, each including how the dataset was composed, the goal for the model and how it can correlate with our approach.

Table 3.1. Learning System Test Scenarios

Scenario	Dataset Provenance	Number of Entries
HVAC System Operation Detection	Sensor Data Collection [**83**]	1000
Water Leak Detection	Master Thesis [**84**]	5607
Agricultural Irrigation Hour	Master Thesis [**85**]	105217
Weather Conditions	Online [**86**]	52609
SmartGrid House Consumption	Online [**87**]	1051200
Communication Strength Signal	Online [**88**]	878289

3.2.3. Results

To train, validate and test the models, in each scenario the presented dataset was divided into three groups: 70% for training, 20% for validation and 10% for testing. For each scenario the results and time of training will be discussed as well as how they are influenced by the size of the dataset.

The accuracy results obtained for each model, for default, hyperparametrization and cross-validation, can be seen in Table B1, B2 and B3, respectively. As can be seen, the accuracy values of each algorithm trained were quite varied, which allowed us to see

which were best suited to the dataset and application under study. Figure 3.7 shows an aggregation of the results from the tables in the appendix.

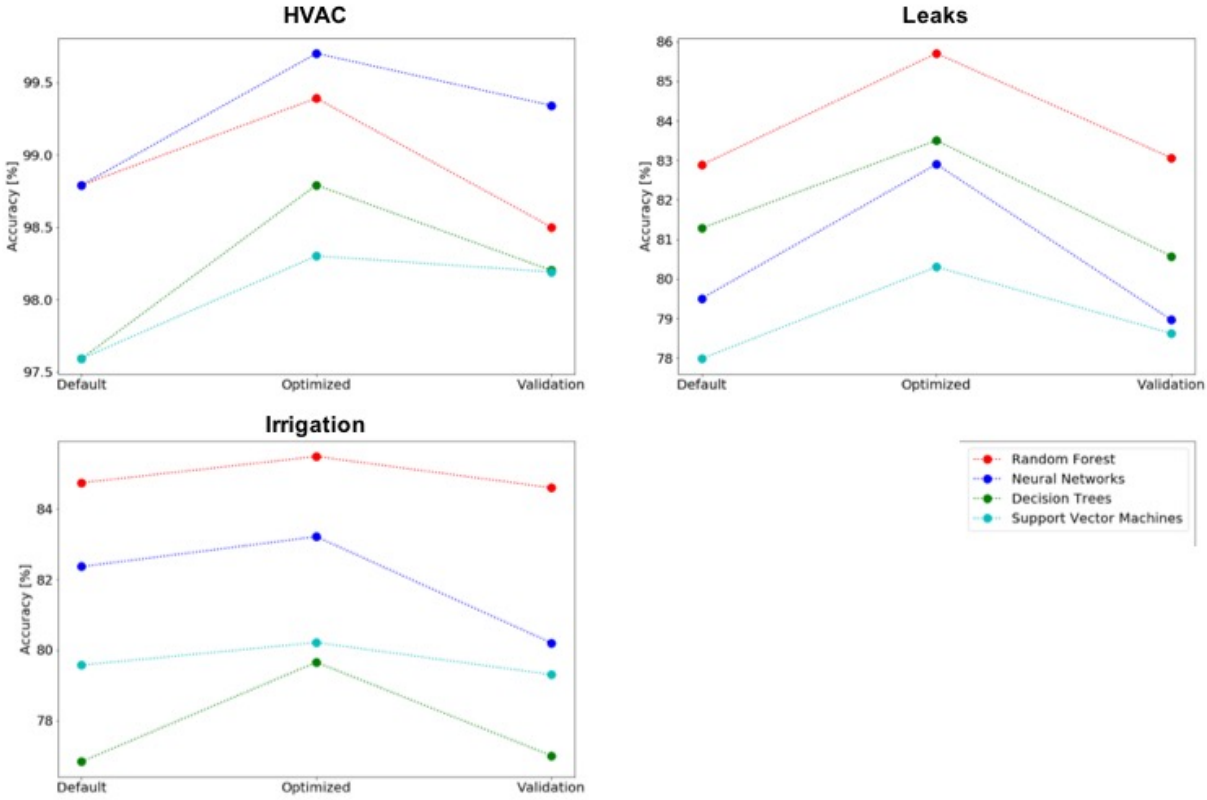


FIGURE 3.7. Classification Results

The first thing to notice, in all models, is that the hyperparametrization values always have the higher accuracy, followed by the cross-validation values and finally the default values. This is justified by the methodology followed, as the default model in the first step is tuned to improve the default result, obtaining always a better result. Then, in the cross-validation step, as multiple combinations of the dataset are tested, and the accuracy for each fold is averaged, it is expected that the accuracy decreases. Although this happens, the cross-validation results are the ones that allow a better knowledge of the model accuracy, as it was exposed to a higher variety of unknown data.

Considering only the cross-validation results, as they are the ones best fitted to evaluate the model accuracy, the best model for classification is Random Forest, the dark blue line in the graph, being the best solution for two out of the three scenarios, by almost 3% in accuracy, and being the second best in the remaining scenario, with under 1% difference in accuracy. The other models have almost a similar pattern among all scenarios, with SVM being the worst solution in all three tests.

Furthermore, this test allows the comparison of the results in terms of the dataset size. Regarding the results from the first tested scenarios, with all models over 98% accuracy and little to none difference between them and between default and cross-validation, it is possible to associate the good results obtained with an overfitted model obtained from the lack of data to test and validate the trained model. The other two scenarios, with larger datasets, showed more diverse results among all models, and also a lower accuracy result.

The size of the dataset also influenced the time it takes to train and generate the best model for each algorithm. Table 3.2 shows the time it takes to train each model for the fully followed methodology, as well as the time to train only the best configuration for the model.

TABLE 3.2. Classification Time Results

Scenario	**Time [min (seg)]**			
	SVM	**Random Forest**	**Neural Network**	**Decision Tree**
HVAC	1 (1)	2 (1)	2 (1)	1 (1)
Leak Detection	6 (3)	5 (3)	8 (5)	2 (1)
Irrigation Hour	350 (180)	250 (64)	320 (180)	3 (5)

As in the classification accuracy the number of entries in the dataset affects the training time. In the first scenario, with the smaller dataset, it was almost instantaneous to train each model, with less than 2 minutes each, and only one second to train the best configuration. The last scenario, with over 100000 samples, takes almost 6 hours to train the model, and 180 seconds to train the best configuration, in the worst scenario.

Comparing each algorithm, it is possible to conclude that the Decision Trees are the ones that take less time to fully train and configure, even with larger datasets, with only 3 minutes needed in the worst scenario. Random Forest is the second quickest to train, on all scenarios, being SVM and NN the worst in all scenarios.

Taking into consideration the accuracy and time results obtained for each scenario, it is possible to establish that Random Forest is the best solution for classification problems.

Continuing our research for the regression scenarios, the MAE results obtained for each model, for default, hyperparametrization and cross-validation, can be seen in Table B4, B5 and B6, respectively. Contrarily to the classification results, that the accuracy is always between 0 and 100%, the MAE results do not have boundaries, and vary from scenario

to scenario based on the data provided, and as such it is harder to compare among them, and to properly display them, two graphs were needed, as shown in Figure 3.8.

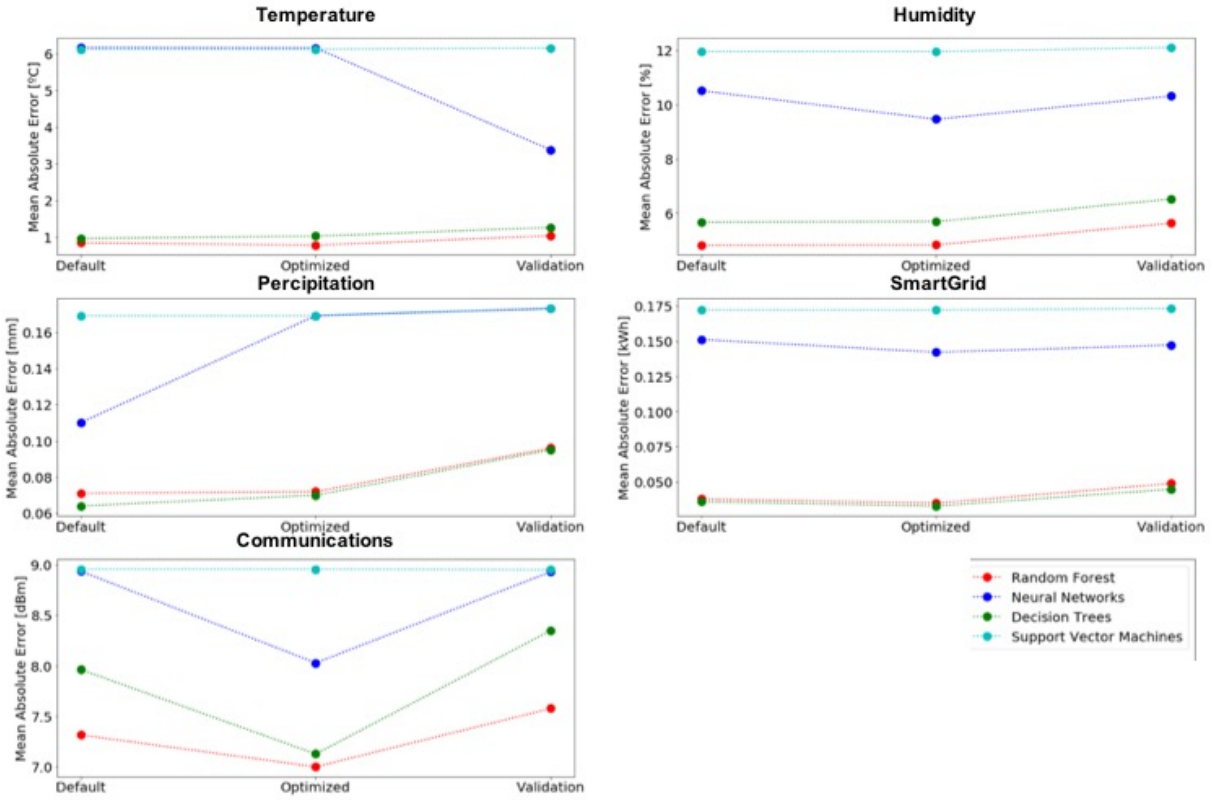


FIGURE 3.8. Regression Results

As in the classification scenario, and since the methodology is the same, only the cross-validation results were considered when comparing the different algorithms. As said, each test has its range of values and measure, so it is harder to compare the results among them. Nevertheless, it is easy to understand that Random Forest is the best overall model for Regression, as it provides the best solution for three out of five scenarios and it is the second best for the remaining two scenarios, in a close margin with Decision Trees. Linear Regression proves to be the worst solution in all scenarios.

Once again, the dataset size influences the model train time, and the obtained results can be found in Table 3.3.

Comparing each algorithm, it is possible to conclude that Linear Regressions are the ones that take less time to fully train and configure, even with larger datasets, with only 22 minutes needed in the worst scenario, although, it also provides the worst results in terms of MAE. Decision Trees and Random Forest are the second quickest to train, with similar results on all scenarios.

Table 3.3. Regression Time Results

Scenario	Time [min (seg)]			
	Linear Regression	**Random Forest**	**Neural Network**	**Decision Tree**
Temperature	2 (2)	11 (15)	4 (5)	12 (8)
Humidity	2 (2)	10 (15)	5 (5)	11 (8)
Precipitation	1 (2)	11 (15)	4 (5)	10 (8)
SmartGrid	22 (10)	34 (25)	125 (120)	30 (20)
Communications	14 (20)	32 (38)	68 (120)	21 (30)

Taking into consideration the MAE and time results obtained for each scenario, it is possible to establish that Random Forest is the best solution for regression problems, with Decision Trees having almost similar overall results.

3.3. Comparison of Edge and Cloud Computing for Data Analysis

With the best supervised learning algorithm selected for both classification and regression, the next goal was to study how these models perform when ported to a Cloud and Edge computing architecture.

The trained models in Anaconda are already ready to port to a Python file that will run in a Cloud Computing script, having the same performance found in the tests done in Section 3.2. But for the Edge Computing script, that will run on the microcontrollers, the model needs to be ported into a C file. Due to the limited Flash in these devices, usually with 8MB or less, the model needs to be adjusted to create a smaller size, and those adjustments can lead to a lower accuracy or higher margins of error.

The goal of this section is to understand the impacts of porting a model to an Edge Computing architecture in accuracy, file size, latency and power consumption needed to perform an analysis.

3.3.1. Accuracy & File Size

To evaluate whether an edge computing approach affects the quality of the model, two of the previously tested models were used, the Irrigation Timing, for classification, and the Communication Signal Strength, for regression. To port the trained model for an edge computing model, the *micromlgen* library [**89**] was used, creating an C file capable of running in the microprocessor.

Regarding the classification model, the best Random Forest configuration achieved used the following parameters:

- Number of Estimators: 212
- Split Criteria: gini
- Minimum Split Samples: 10
- Maximum Depth: 196

This model achieved a 85.49% accuracy and file size, after porting, of 559MB. As discussed, to be able to run in a microprocessor, which normally has up to 8MB of Flash Memory, the ported model needs to be under 4MB, so the model needs to be optimized.

First, the features type was evaluated and adjusted, in order to reduce the amount of space needed to store and analyze them. Table 3.4, shows the initial and adjusted data type.

TABLE 3.4. Irrigation Time Features Data Type

Feature	**Initial**	**Adjusted**
Year	int64	int8
Month	int64	int8
Day	int64	int8
Hour	int64	int8
Temperature	float64	float32
Relative_Humidity	float64	float16
Total_Precipitation_Low	float64	float32
Wind_Speed	float64	float16
Wind_Direction	float64	float32
Soil_Humidity	float64	float16
Had_Irrigation	int64	int8
Need_Irrigation	int64	int8
Is_Favorable	int64	int8
Suggested_hour	int64	int8

After adjusting the features data type, the model was retrained and ported, achieving an accuracy of 85.49% accuracy and file size of 559MB, exactly the same results as before. It is possible to conclude that the feature data type does not influence either the accuracy or size of the model.

Following with the evaluation, the number of trees in the model, i.e. number of estimators, was the next configuration tested. The goal is to decrease the number of estimators and check the impact in both accuracy and size of the model. Table 3.5 shows the obtained results.

TABLE 3.5. Estimators Impact on Ported Classification Model

Feature	**Results**				
Estimators	100 (-52.83%)	50 (-76.42%)	25 (-88.21%)	10 (-95.28%)	5 (-97.64%)
Depth	196 (0.00%)	196 (0.00%)	196 (0.00%)	196 (0.00%)	196 (0.00%)
Accuracy [%]	85.33 (-0.19%)	85.17 (-0.37%)	85 (-0.57%)	84.42 (-1.25%)	83.09 (-2.81%)
Size [MB]	263 (-52.95%)	132 (-76.39%)	66.1 (-88.18%)	26.8 (-95.21%)	13.3 (-97.62%)

It is possible to determine that reducing the number of trees, or estimators, in a Random Forest model will create a file that is almost proportionally reduced in terms of size, with half the trees creating half the size and with less 97% of estimators the output file is 97% smaller. It is also possible to assess that the reduction of estimators does not have a major effect on the model accuracy, with only a 3% decrease in accuracy when using 97% less trees. As such, reducing the amount of trees in a Random Forest model is a good way to reduce the file size for an edge computing model without compromising its accuracy and efficiency.

Although reducing the number of estimators can reduce proportionally the size of the model, without a major effect on the accuracy, the size of the model is still too big to work on a microprocessor. Without the possibility to reduce even more the number of estimators, the depth of each tree was the next parameter to assess on how it impacts the model accuracy and size. As for the number of estimators, the reduction of depth was performed on the best model, i.e., without changing the number of estimators or using the best result from the previous test. Table 3.6 shows the obtained results.

Contrarily to the estimator approach, reducing the depth of each tree in the Random Forest model does not proportionally affect the file size. Although, when a smaller depth is used, the file size can be reduced by up to 99%. Despite that capability of reducing the file size, when the depth of the model is reduced it has a higher impact on model accuracy, with a 10% lower result, compared with the best model, which is 7% higher than when reducing the number of estimators, for a similar file size output. As such, to

Table 3.6. Depth Impact on Ported Classification Model

Feature	**Results**		
Estimators	212 (0.00%)	212 (0.00%)	212 (0.00%)
Depth	20 (-89.80%)	10 (-94.4%)	5 (-97.45%)
Accuracy [%]	85.32 (-0.20%)	80.97 (-5.29%)	76.59 (-10.41%)
Size [MB]	494 (-11.63%)	43.3 (-92.25%)	1.62 (-99.71%)

reduce the model size, reducing the depth of a Random Forest is equally good as reducing the number of estimators, but it has a higher influence on model accuracy and efficiency.

Retraining with a new depth configuration allows the model to run in a microcontroller but also has a higher impact on the accuracy, when compared to the estimator approach. As reducing the number of estimators allows for maintaining the model accuracy and reducing the depth allows for a smaller file, an approach where both values are reduced can result in a small file without a higher drop in accuracy. For that, the best configuration from both approaches were combined, and the result can be found in Table 3.7.

Table 3.7. Hybrid Approach Impact on Ported Classification Model

Feature	**Results**					
Estimators	25 (-88.21%)	10 (-95.28%)	5 (-97.64%)	25 (-88.21%)	10 (-95.28%)	5 (-97.64%)
Depth	10 (-94.90%)	10 (-94.90%)	10 (-94.90%)	5 (-97.45%)	5 (-97.45%)	5 (-97.45%)
Accuracy [%]	80.9 (-5.37%)	80.47 (-5.87%)	80 (-6.42%)	80.17 (-6.22%)	74.48 (-12.88%)	75.25 (-11.98%)
Size [MB]	5.14 (-99.08%)	2.09 (-99.63%)	0.995 (-99.82%)	0.192 (-99.97%)	0.0773 (-99.99%)	0.0388 (-99.99%)

Following an hybrid approach of reducing both estimators and depth it is possible to obtain a model file with an acceptable size and with a smaller loss in accuracy, when compared with the depth approach. Using 95% less estimators and depth is possible to achieve a 99% smaller file, almost the same as in the depth approach, with only a 5% decrease in accuracy, which is 5% less lost than in the depth approach. If an even smaller file is needed, re-adjusting the configuration, it is possible to achieve a 99.99% smaller file with a 6% decrease in accuracy.

Thus, it is possible to conclude that tweaking these parameters allows for a fully trained model to be ported to a microcontroller without compromising its integrity, and that it can be adjusted to fulfil the best scenario needed.

To understand if this approach has the same results on both classification and regression models, it was applied to the previously trained Communication Signal Strength scenario. For that, the best Random Forest configuration was achieved with the following parameters:

- Number of Estimators: 200
- Split Criteria: mse
- Minimum Split Samples: sqrt
- Maximum Depth: 63

This model achieved a MSE of 7.187 dBm and file size, after porting, of 3843MB. As the previous model, this size is too large to run on a microcontroller.

Using the same approach, of reducing both the number of estimators and depth, the MSE and file size obtained are presented in Table 3.8.

As in the classification model, the Random Forest regression follows the same path when reducing the number of estimators and depth, with a 99% smaller file when using 95% less estimators and depth. In this scenario, there was not an accuracy to compare and to model between 0-100%, since MSE is a real number ($\mathbb{R}$), so only the variation of the MSE, facing the best value, was accountable. As such, with a 99% smaller file a 27% higher variation of MSE was obtained, accounting for almost 2 dBm, which is still an acceptable value for the regression. As in the previous test, if a small file is needed, tweaking with the configurations is possible to achieve a 99.999% smaller file, compromising only more 4% of the MSE.

This shows that the approach for creating edge computing models from cloud computing models is possible and that can be applied for both classification and regressions,

TABLE 3.8. Hybrid Approach Impact on Ported Regression Model

Feature	**Results**					
Estimators	25 (-88.21%)	10 (-95.28%)	5 (-97.64%)	25 (-88.21%)	10 (-95.28%)	5 (-97.64%)
Depth	10 (-94.90%)	10 (-94.90%)	10 (-94.90%)	5 (-97.45%)	5 (-97.45%)	5 (-97.45%)
MSE [dBm]	9.131 (-27.05%)	9.137 (-27.13%)	9.104 (-26.67%)	9.422 (-31.10%)	9.435 (-31.28%)	9.44 (-31.35%)
Size [MB]	7.15 (-99.81%)	2.85 (-99.93%)	1.4 (-99.96%)	0.182 (-99.995%)	0.0766 (-99.998%)	0.0389 (-99.999%)

using Random Forest, creating small files that mimic the model, and run inside microcontrollers, without compromising its accuracy. Although some quality is lost, it is important to notice that this will allow for systems to decide in real-time, without depending on third-party computing and exchange of messages, and the latency involved, being an excellent solution for non-critical decisions or for solutions without network capabilities or running on batteries. Also, these edge computing decisions can always be backed with a cloud computing analysis when needed.

3.3.2. Analysis Time & Power Consumption

As described in the previous section, Edge Computing allows the decision to be made in real-time directly in the device, without further messages or third-party decisions, allowing the reduction of latency in the decision. Our research showed that the Edge Computing models have a 6% decrease in accuracy, when facing the same Cloud Computing model, due to the need for the model to be ported to a smaller file, which can only be done by adjusting the model parameters, such as depth and number of estimators.

To understand if this decrease in accuracy can be compensated by the decrease in latency, that allows for quicker decisions, or the decrease of exchanged messages, that allows for less energy being used in the node, the Irrigation Timing model, used in the last section, was implemented, in both Cloud and Edge Computing, to evaluate the analysis time and power consumption of each model, as well as the accuracy of the model. In

Table 3.9 the obtained accuracy and file sizes for the Cloud and Edge Computing models are shown.

TABLE 3.9. Irrigation Timing Model Specifications

Model	**Accuracy [%]**	**File Size [MB]**
Cloud Computing	85.49	559
Edge Computing	80.47	2.09

To deploy the models two nodes were simulated using the ESP32-DevKitC-v4, a development board that includes the ESP32 microcontroller, 4MB of Flash and WiFi connectivity. In the first, the Cloud Computing methodology was implemented, being the node responsible for sending a request to the cloud for the model result and waiting for the response, whereas, in the second node, the Edge Computing methodology was implemented, being the model decision made directly on the node. On both nodes, the necessary data for each model analysis, the sensor data, was simulated using random values that fit the normal values found in the used dataset.

The simulation for each model was performed 50 times and for each of these decisions, the accuracy, the time elapsed between the request and the result, the average power consumption in this time and the number of messages exchanged were evaluated. Table 3.10 shows the obtained results.

TABLE 3.10. Analysis Time & Power Consumption Results

Model	**Accuracy [%]**	**Average Time [μs]**	**Average Energy [mA]**	**Average Messages Exchanged**
Cloud Computing	83.31	238153	122	2
Edge Computing	78.93	313	57	0

In terms of accuracy, it is possible to conclude that in both scenarios the behavior is the same, both decreasing around 2% in accuracy, so, when considering only the model accuracy, Cloud Computing has an advantage, as already concluded before.

It is on the other results that Edge Computing shows its advantages and compensates for the lower accuracy. In terms of latency, the time between the decision being asked and the result being available, although Cloud Computing needs only 238 ms, Edge Computing can achieve a result in under 1 ms, needing only 313 μs, almost 760 times faster than Cloud Computing. This is justified by the messages exchanged in both scenarios, as Edge computing requires no messages, since the analysis is done directly on the node, and Cloud

Computing requires a request being done by the node to the cloud and then waiting for the result to be sent from the cloud. This proves that, although some accuracy is lost, Edge Computing allows for a quicker response time, that can be crucial in systems that require near real-time decisions, and that are non-critical in terms of decisions, allowing for some margin of error.

Edge Computing also takes advantage in terms of energy needed to get a decision. On average, the model requires 57 mA to get a decision, while the Cloud Computing needs 122 mA, more than 50% when facing Edge Computing. Once again, this is justified by the lack of messages needed by the Edge Computing model, with external communications, via WiFi, Cellular, or other methods, usually accounting for the most power consuming activity in an IoT node. With this, it is possible to conclude that, when using an Edge Computing approach, a more sustainable and low-power node can be achieved, allowing for them to run on batteries for longer, and also to be able to introduce intelligence in areas without network capabilities.

Once again, Edge Computing can always be backed with a Cloud Computing approach when a more critical decision is needed.

3.4. Remarks

This chapter establishes the basis for the importance of the learning systems in the developed solution. From the introduction of the concepts to its research and implementation, it presented a detailed approach of creating and evaluating learning systems and porting them to their computing location.

With the goal of assessing the best Machine Learning algorithms to use for data analysis in IoT projects, several were studied, among them Decision Trees, Neural Networks, Linear Regressions and SVM, although it was Random Forest the one with the best results for both classification and regression. This research also pointed to the importance of analysing several models, as each scenario could benefit from a specific model. Finally, configuring and validating the models proved to be essential to achieve the best results and to guarantee the efficiency of the model outside the training environment.

Being this a solution that will depend on edge nodes, capable of running some intelligence directly on them, a study to evaluate how this computing methodology would affect the efficiency of the models was also performed. It was possible to conclude that porting the fully trained models to an edge device implies some changes into its configuration, in order to create a small model capable of running on low memory devices, that

affect its performance, reducing accuracy of the original model. Nevertheless, with the proper configuration of these parameters, it is possible to create a smaller model without compromising entirely its efficiency, allowing a proper edge device analysis.

The advantage of Edge Computing was proven by the reduced latency and power consumption when facing a Cloud Computing approach, being able to assess that the lower accuracy is widely compensated by these, as it provides for faster decisions and more sustainable nodes.

With the research presented in this chapter, several key points were gathered for the development of the entire system, as this will be supported by Machine Learning techniques, allowing not only an easier development of the next phases but also to guarantee consistency and reliability in those phases.

CHAPTER 4

Autonomous Communication System Configuration

This chapter presents the research and developed solutions for the autonomous configuration of the communication systems. It starts with an overview on the research, containing the related work found in the literature, followed by an explanation of some of the most important definitions, including all the communication protocols used as well as an introduction to peer-to-peer and cloud communication. With all the definitions introduced, this chapter continues with the research done to find the best communication protocol for each of the point-to-point communication, detailing the methodology followed, the scenarios and tests performed and finally the obtained results and remarks. It follows with the developed configuration models for the communication systems, both point-to-point and cloud, including a comparison of an edge and cloud computing approach. Finally, a remarks section closes this chapter with a brief discussion and conclusion of the topics and results given in this chapter.

4.1. Overview

Communication is a major part of IoT systems and with the constant evolution of devices and solutions more and more communication protocols arise, existing nowadays more than 20 different protocols that can be used to connect devices. In an IoT system messages can be exchanged inside the network, between the several nodes deployed, or between the nodes and cloud servers. But as the name describes, IoT relies on the ability to have a way to communicate between the "things" and the "Internet" and that is where the main issues start to rise. One of the biggest problems is the lack of coverage or reliability of communication protocols in certain environments, mainly when comparing indoor and outdoor projects or city versus rural environments. In a Smart City environment, a WiFi connection can be easy to find, a LoRaWAN gateway is available or a strong 4G signal covers the entire area, but when it moves to rural areas, underdeveloped cities or poor regions, the availability of these network structures can be weak or non-existent.

So, in order to deploy an autonomous communication configuration system that can truly adapt to any specification or environment, the nodes need to adjust to the local

characteristics and not the other way around. In other words, the node has to communicate with the available networks and not demand the installation of networks to be able to work.

With Machine Learning being more and more used alongside wireless communications, the literature presents several researches, including a great survey [**90**], covering several applications from security, interference, link configuration and node locations.

Our goal to create an autonomous communication configuration system 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 [**91**], 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 [**92**], using Dynamic Selection, with 96% efficiency, and [**93**] 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 [**94**].

As our research is focused on creating a modular IoT solution capable of adapting to any situation, based on Machine Learning, several approaches in terms of network protocols need to be available, both for point-to-point communication as for cloud communication.

4.1.1. Point-to-Point Communication

As said, device to device communication is a major part of IoT systems and new advances in technology introduced wireless networks developed solely for IoT projects or low-power devices, such as LoRa, BLE or ZigBee, with Wi-Fi still having a major contribution and being modified to fit these new specifications. Since each project might need a specific protocol, due to location, implementation conditions or energy supply, the developed system will be available to work with the following technologies.

4.1.1.1. *BLE.* Bluetooth Low Energy (BLE) is an upgrade on the Bluetooth technology, designed to consume the least amount of energy while using the same wireless standard

[**95**]. BLE, working on the same 2.4GHz as Bluetooth, reduced the high speed and high rate transmission, allowing for a decrease in power consumption by up to 80% and increasing the range by 10 times, with connections up to 100 meters [**96**]. BLE also introduced mesh and star topologies, in a one-to-many fashion, contrary to the simple one-to-one connection provided by classic Bluetooth. As BLE is designed to broadcast short messages in close spaces, it becomes one of the most used technologies in IoT projects.

4.1.1.2. *ESP-NOW.* ESP-Now is a peer-to-peer wireless protocol developed by Espressif which enables multiple devices to communicate with one another without using Wi-Fi. The pairing between devices is needed prior to their communication, so after pairing a device with each other, the connection is persistent [**97**]. 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, being more power-efficient and faster to deploy when compared to Wi-Fi [**98**], supporting up to 20 nodes and being limited to 250 bytes packets.

4.1.1.3. *LoRa.* LoRa is a long range low power wireless technology that uses unlicensed radio spectrum, usually on the 868 or 915 MHz range, based on Chirp Spread Spectrum (CSS) modulation to allow for the communication reach [**99**]. As it aims to eliminate repeaters, reduce device cost, increase battery life and support a large number of devices, it is the perfect solution for most IoT projects that rely on gathering data on large areas with low-power devices [**100**]. These features are possible since LoRa works on a star topology, reducing complexity and congestion in the network, allowing for a viable low power long communication, with a single gateway covering up to hundreds of square kilometers [**101**].

4.1.1.4. *ZigBee.* ZigBee enables low-cost, low-power and low-data rate Machine-to-Machine communications for IoT networks [**102**], in the 868 MHz, 915Mhz and 2.4GHz frequency bands. Based on the IEEE 801.15.4 physical layer and the Medium Access Control sublayer, it is complemented by an application framework layer defined by the ZigBee Alliance [**103**]. Capable of working in mesh, star or cluster topology, ZigBee can achieve distances up to 100 meters, when working on the 2.4GHz frequency, or up to 1 kilometer, when using the lower frequencies [**102**]. ZigBee is the preferred solution for smart homes solutions, due to its low power capabilities.

4.1.2. Cloud Communication

Besides the intra network communications, devices might need to exchange messages to cloud servers, for analysis, storage or even communication with other networks. For years this communication was performed solely over Wi-Fi, Ethernet, cellular networks or even satellite. 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. Since each project might need a specific protocol, due to location, implementation conditions or energy supply, the developed system will be available to work with the following technologies

4.1.2.1. ***Wi-Fi.*** Based on the IEEE 802.11 standard, Wi-Fi is the most common way for devices to connect wirelessly to local networks and to the internet. Operating in the 2.4 GHz and 5 GHz bands, it can connect devices either in ad hoc, a peer-to-peer connection such as ESPNow, or access point mode, where devices connect to an access point (AP) [**104**].

4.1.2.2. ***LoRaWAN.*** LoRaWAN is built on the lower level of the LoRa protocol, mainly on the physical level [**105**]. Contrarily to LoRa, which is a point-to-point communication protocol, LoRaWAN messages are received by all the base stations in range, being the back end responsible for removing duplicate receptions, checking security, sending acknowledgments to the end device, and sending the message to the corresponding application server [**106**]. The characteristics combined with the ability for users to deploy gateways, makes LoRaWAN suitable and more robust for city-scale IoT deployments [**106**].

4.1.2.3. ***SigFox.*** Sigfox is an LPWAN network that deploys its proprietary base stations equipped with cognitive software-defined radios and connects them to the back end servers using an IP-based network [**106**]. SigFox uses Differential Binary Phase-Shift Keying (D-BPSK) modulation to send messages up to 12 bytes, on the 868 MHz radio spectrum, in Europe, with a speed of 100 bps. This modulation technique is part of the Ultra-Narrow Band (UNB) modulations category, like the CSS used in the LoRa modulation scheme, with the advantages of being more efficient, receive radio signal below the noise level, close to -145 dBm, being easy to implement and transmit data in a low bit rate, reducing the costs of the hardware components [**107**]. The disadvantage of SigFox is that number of messages over the uplink is limited to 140 messages per day and the number

of messages over the downlink is limited to four messages per day, which means that the acknowledgment of every uplink message is not supported [**106**].

4.2. Comparison of Point-to-Point Protocols for Low-Cost IoT Devices

In the literature it is possible to find several studies comparing wireless communication protocols, but none focusing on multiple outputs or scenarios and each one using a different methodology, making it almost impossible to compare among them which is the best overall protocol.

Despite those studies, that individually provide good research when focusing on a particular field or output, our goal is to analyse the same protocols under the same methodology, for multiple scenarios, in order to understand which is the one that has an overall advantage or how multiple protocols can be used in a modular and adaptable solution.

For that, the chosen protocols were tested under different conditions and specifications to analyse range, power consumption and reliability.

4.2.1. Methodology

As said, communication is one of the major components of an IoT system, and as these system are usually composed by WSN with multiple nodes interchanging messages, a reliable protocol must be used to ensure the best communication between nodes, not only in terms of quality of signal but also in terms of reliability, range and power consumption.

To evaluate which is the best communication protocol to transmit data between two devices inside a sensor network under several scenarios and specifications, two nodes were used to send a message between them using the communication protocols under study. For each scenario, several specifications were tested, such as the presence of obstacles, distance, transmission power, among others.

For that, several wireless communication protocols were used, being their characteristics displayed in Table 4.1.

On each test scenario, each protocol was tested with the presented power transmission, to understand how that can affect the performance of the protocol.

TABLE 4.1. Point-to-Point Protocols Characteristics

Protocol	**Transmission Power** [dBm]	**Energy Consumption** [mA]			**Range** [m]		**Sensitivity** [dBm]
		Transmit	**IDLE**	**Sleep**	**Indoor/ Urban**	**LoS/ Rural**	
BLE [**108**]	-7	8	10	0.07	100	250	-94
	14	10					
ESP-Now [**108**]	-	90	60	-	100	500	-98
RF [**109**]	14	50	16	0.001	200	500	-120
	20	150					
LoRa [**110**]	5	20	12	0.001	500	2000	-135
	23	120					
ZigBee [**111**]	-7	20	17	0.002	60	1200	-103
	8	40					

Note: All the values are based on the used modules for the test, identified by the reference in the protocol column

4.2.2. Test Scenarios

The modularity of the proposed system allows it to be adapted to a wide range of solutions and specifications, meaning it needs to adapt to multiple scenarios and applications and that communication must always work, either indoors or outdoors, in a rural or urban environment.

Range figure estimates are based on free-air terrain with limited sources of interference. Actual range and reliability will vary based on transmitting power, orientation of transmitter and receiver, height of transmitting antenna, height of receiving antenna, weather conditions, interference sources in the area, and terrain between receiver and transmitter, including indoor and outdoor structures such as walls, trees, buildings, hills, and mountains.

As such, and to have a better understanding of which protocols work better in as much situations as possible, or if different approaches for each scenario are needed, an evaluation of point-to-point communications protocols was performed for multiple specifications, for both indoor and outdoor, with line of sight and obstacles, and also some of the most usual implementations of IoT systems in Smart Cities.

The chosen scenarios are described in the next sections, including the main goal for that particular test case.

4.2.2.1. *Indoor 1 - Line of Sight + Obstacles.* The first scenario was performed indoors, on a 150 meters straight corridor inside the Building II at ISCTE campus, since it was the longer straight space found that had doors or other obstacles to create interference in the line of sight. Figure 4.1 shows both ends of the test site.

FIGURE 4.1. Indoor Line of Sight Scenario

In this scenario the tested distance between the nodes was 1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 120 and 140 meters. For the obstacle test the same distances were used, but the door in the middle of the corridor was closed and the transmitting and receiving nodes were equally distant from the door.

4.2.2.2. *Outdoor 1 - Line of Sight + Obstacles.* The second scenario was performed on Parque Eduardo VII, in Lisbon, an outdoor location with a line of sight of over 500 meters, as shown in Figure 4.2. Parallel to the line of sight there is a line of trees that were used for the obstacle test.

In this scenario the tested distances between nodes were 1, 5, 10 meters and then increments of 10 meters until the 100 meters mark, followed by increments of 20 meters until the 300 meters mark, and finally by 50 meters increments until the 500 meters mark. The same were used in the tree line to have obstacles between the nodes.

4.2.2.3. *Indoor 2 - Household Environment.* For the second indoor scenario an 135 m^2 apartment was used, with several points tested around one transmitting node, including points in the same division or across several divisions, to have walls, doors and furniture as obstacles. Figure 4.3 shows the location of the transmitting node, the red dot, and the test locations, the green dots, in a grid system.

4.2.2.4. *Outdoor 2 - Urban Environment.* In the last scenario, an urban environment was tested, with several points around one transmitting node, covering an area

(a) (b)

FIGURE 4.2. Outdoor Line of Sight Scenario: a) Google Maps Overview, b) Test Location

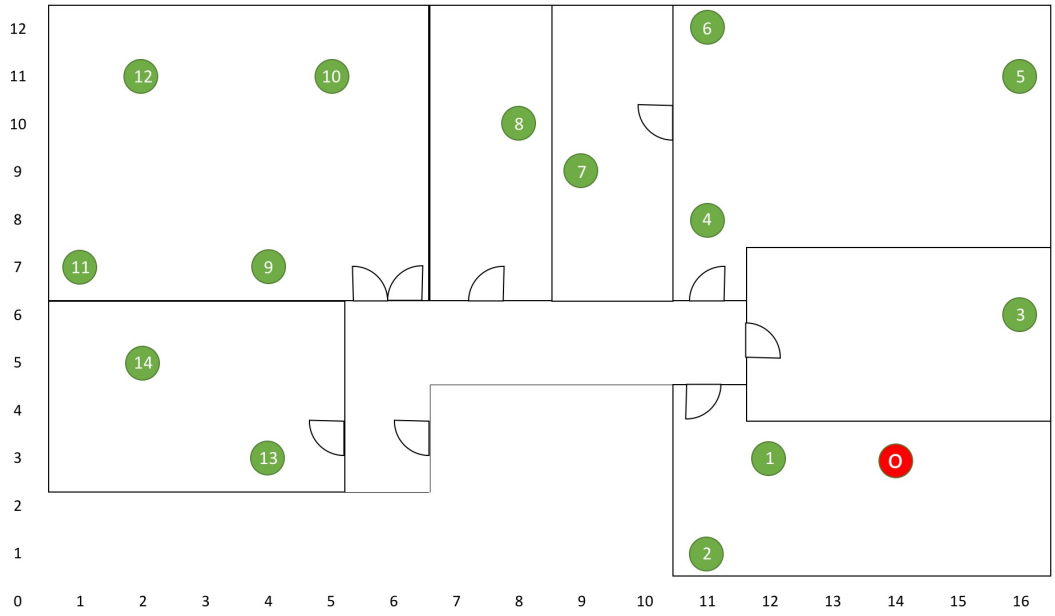
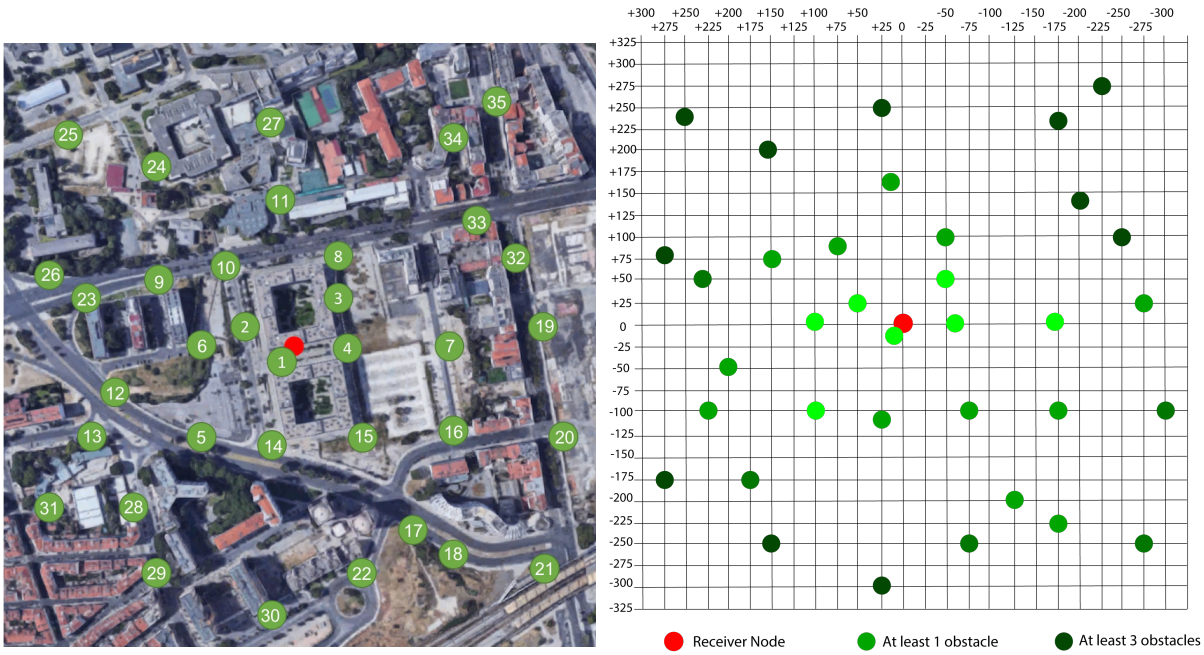


FIGURE 4.3. Household Scenario

of 36ha, with a radius of 600 meters from the transmitting node position, as shown in Figure 4.4 (a), where the red dot represents the transmitting node and the green dots represent the test locations. Figure 4.4 (b) represents the specifications of each test location, with grid position (X,Y), distance and number of obstacles.



(a) (b)

FIGURE 4.4. Urban Scenario: a) Device Locations, b) Locations Information

4.2.3. Results

Figure 4.5 shows the obtained RSSI values for the transmissions in the first indoor scenario, both the line of sight and obstacle line.

It is possible to check that almost every protocol was able to transmit in the full distance of the scenario test, the 150 meters, on all transmission power configurations. The only exception was BLE, which only reached that with the higher settings, being the lowest transmission power, -7 dBm, only capable of reaching 60 meters and the middle setting, 3 dBm, only capable of 80 meters.

When analysing the obstacle results, a totally different result is found. Only RF and LoRa were able to reach the full 150 meters range, even with the lowest settings, whereas ESPNow, BLE and ZigBee were only capable of reaching 40 meters with the lowest settings and 120 meters for BLE and ZigBee with the highest transmission power. This is an indication that RF and LoRa handle better with obstacles in the transmission path. This can be justified by the use of lower transmissions frequencies and the CSS modulation.

Figure 4.6 shows the obtained results for the first outdoor scenario, both the line of sight and obstacle line.

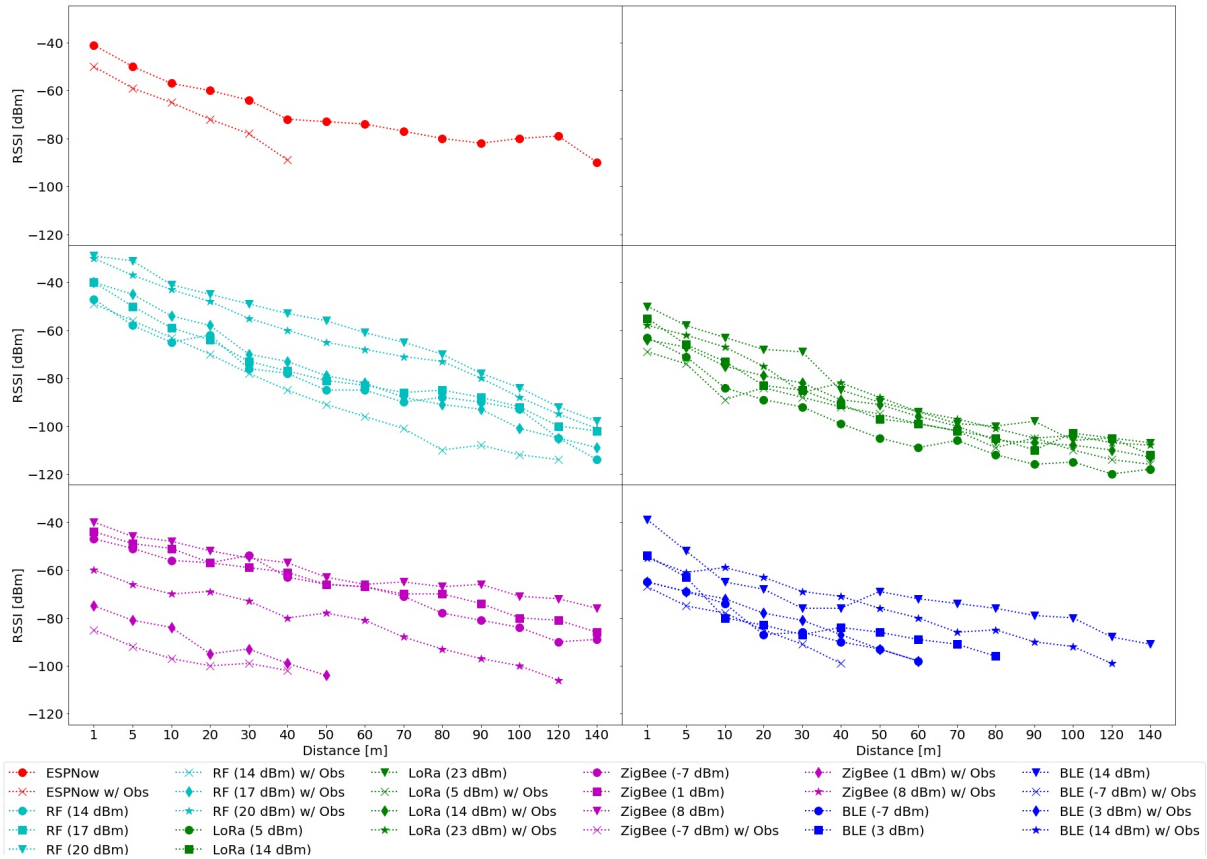


FIGURE 4.5. RSSI Indoor Line of Sight Results

In this scenario it is possible to understand the true limitations, in terms of range, of the protocols with only LoRa and ZigBee being able to reach the 500 meters transmission mark. Even so, it was only possible with the full transmission power, with ZigBee only reaching 200 meters and LoRa 240 meters, with the lowest transmission power. ESPNow only reached 180 meters, RF 220 and 140 meters, and BLE 140 and 50 meters, for the highest and lowest settings, respectively.

In terms of obstacle interference, LoRa proves again to be the best protocol, reaching the 500 meters mark with the highest transmission power and 180 meters with the lowest. ZigBee showed the worst discrepancy between line of sight and obstacles, reaching only 100 meters with the highest settings and 50 meters with the lowest.

On a straight line, LoRa proved to be the best solution to reach further distances and compensate for the interference of obstacles, in both indoor and outdoor scenarios. Even so, LoRa is also the second most power consuming protocol under test, so when lower distances are needed or no obstacles exist, ZigBee can be a better alternative in terms of power consumption and reliability.

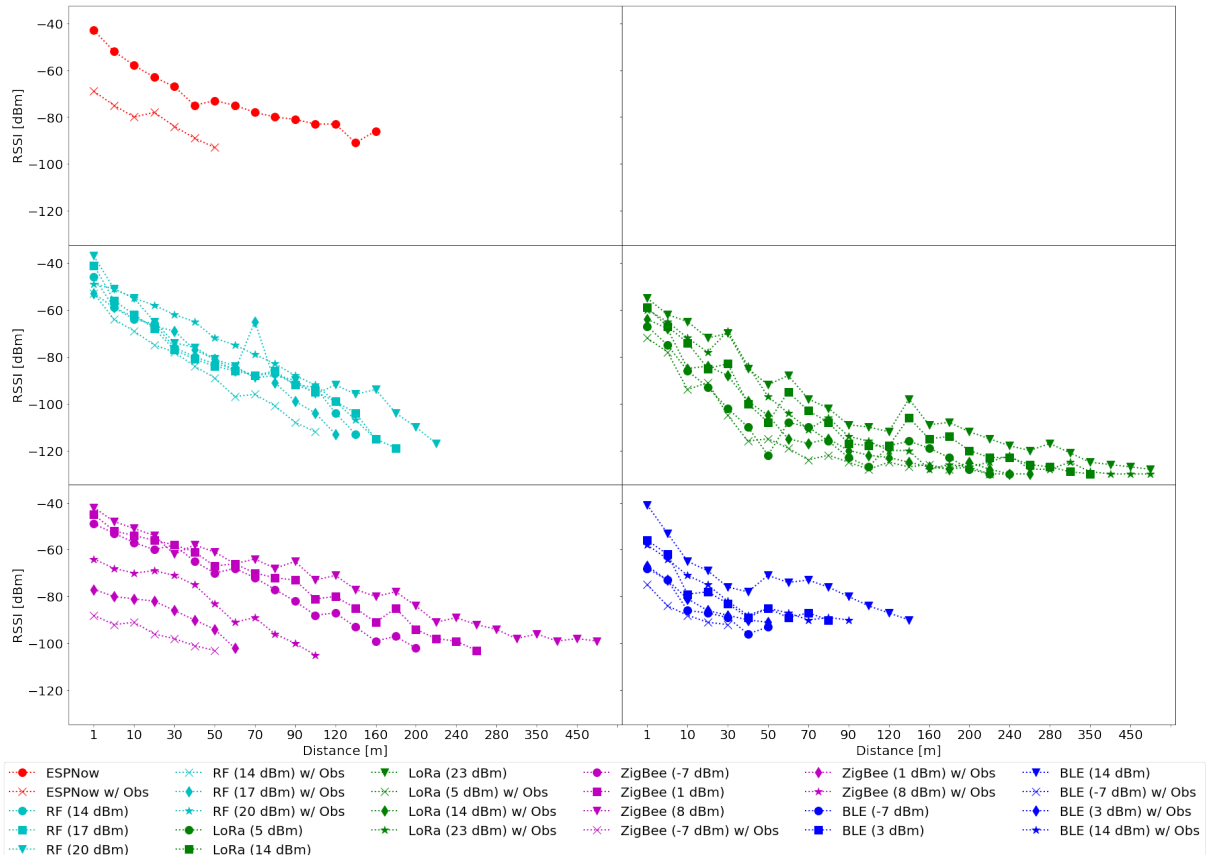


FIGURE 4.6. Outdoor Line of Sight Results

Since a straight line of sight is not a usual scenario, the remaining test, in an urban and residential scenario, might help understand how these protocols will really adapt in a real case scenario. Contrarily to the previous scenarios, which evaluate how far the link was able to create a connection with or without obstacles, in these next scenarios, the goal is to understand how the location specification, in terms of distance, number of obstacles and location affect each protocol.

Figure 4.7 shows the obtained results for the household environment scenario.

It is possible to check that RF and LoRa were able to create a reliable link in every tested location and with every transmission power, while ESPNow, BLE and ZigBee had some difficulties in the further distance with multiple obstacles, even when working with the highest transmission power. ZigBee, once again, proved that it is not very tolerable to obstacles, being the protocol with more unavailability across all locations. The same can be applied to BLE when working with the lower transmission power.

Finally, Figure 4.8 shows the obtained results for the urban environment scenario.

LoRa emerges, once again, as the best protocol, being the only one capable of creating a communication link in more than half of the locations and the only capable of reaching

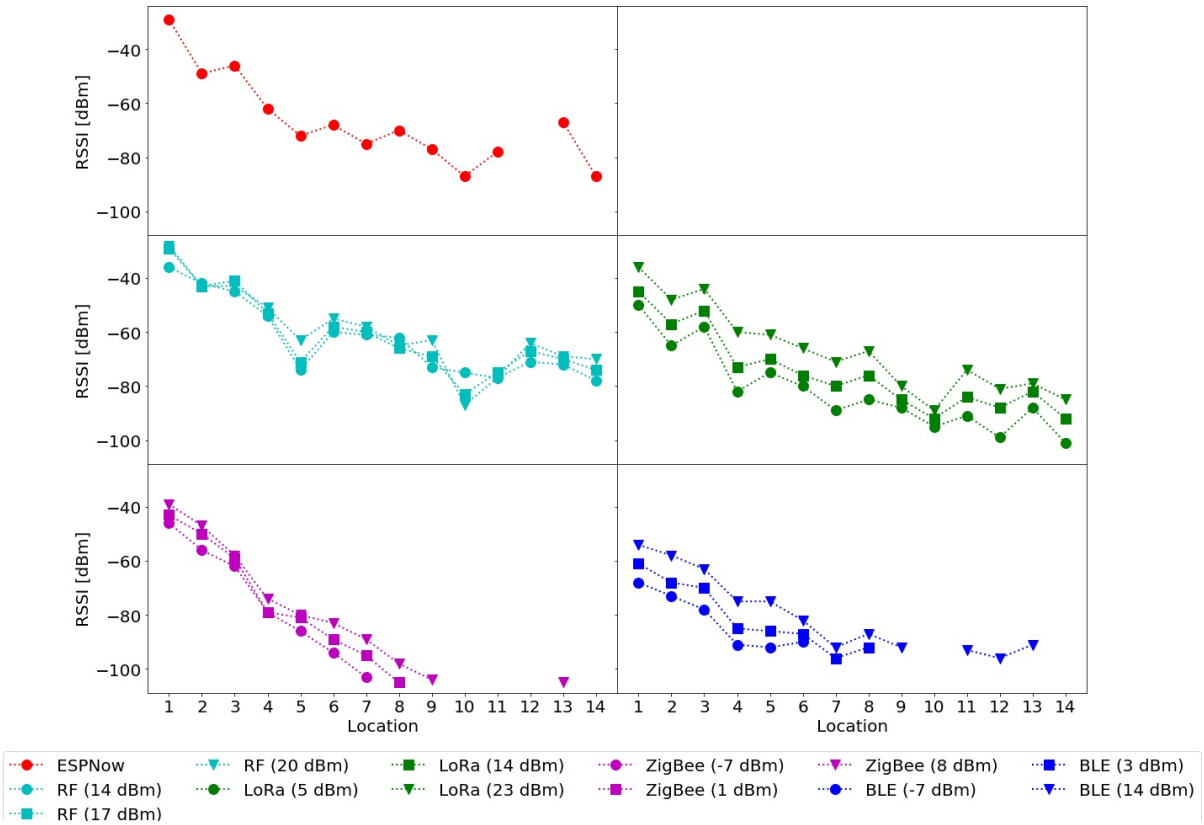


FIGURE 4.7. Household Environment Results

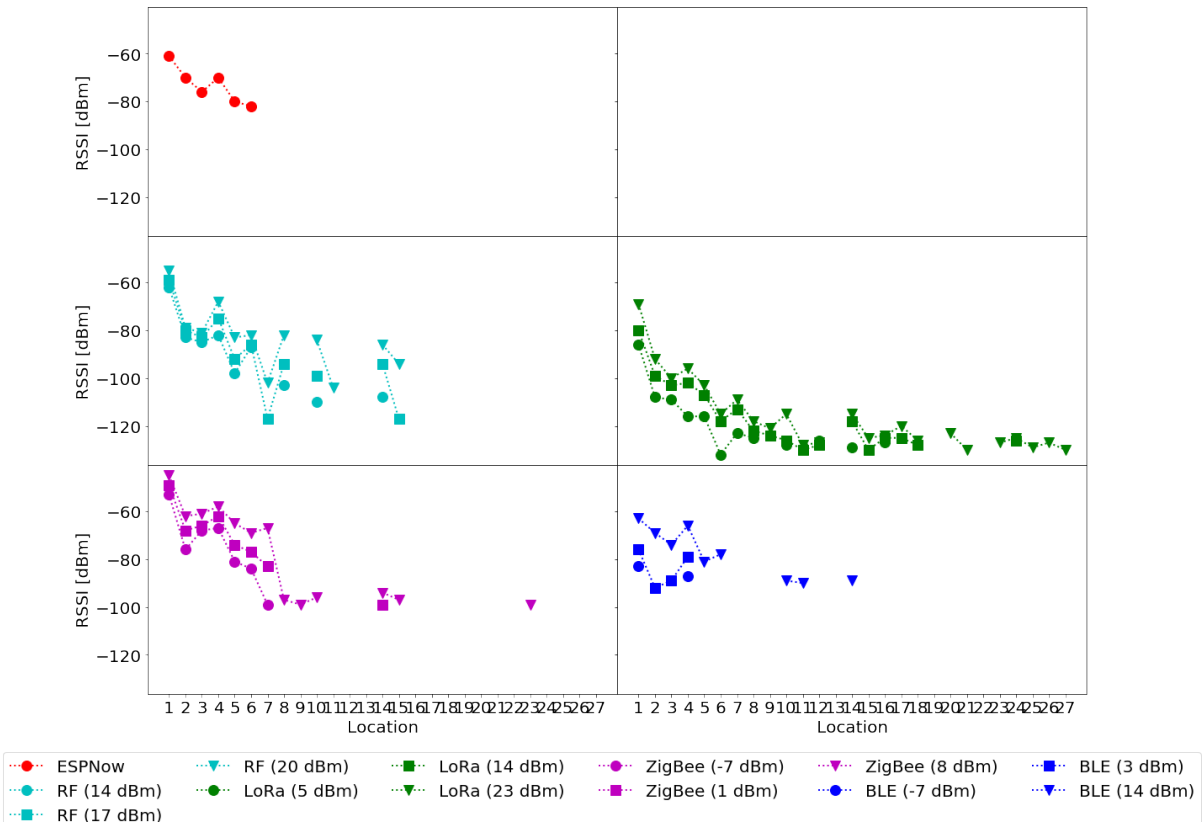


FIGURE 4.8. Urban Environment Results

the further locations tested. Nevertheless, it also shows some limitations, mainly when more than three buildings are in between the nodes. ESPNow and BLE proved not to be a choice when an outdoor urban environment is the application site, with less than 20% of the locations being able to connect, even with the best settings. RF and ZigBee were able to create connections with the closest nodes, but had no link connection when obstacles were presented.

Following the straight line tests results, LoRa continues to have the best performance in all scenarios, and even with the second most power consuming module, it proves to be the best overall solution for point-to-point communications between IoT nodes. For indoor locations or situations with a low possibility of obstacles, ZigBee can be a great solution, since it can match the range of LoRa while reducing the power consumption in almost half.

Another conclusion that is possible to draw from this study is that similar results in terms of availability, reliability and range can be obtained while using lower transmission power configuration that reduces the amount of energy needed to exchange messages.

4.3. Point-to-Point Communication Configuration System

As proved by our point-to-point communication test, using a higher transmission power value does not always result in a better communication link, since lower values can reach the same distances while saving power and since multiple nodes transmitting in full power can generate interference in the network [**100**]. This shows that not only is it possible to reduce the energy consumption of the device, but also improve the network reliability, by adjusting the transmission power of each end node.

This section presents a methodology for an implementation of an autonomous configuration system for peer-to-peer communication in smart nodes supported by Machine Learning, that uses regressions to predict the energy consumption and link quality of a connection and then chooses the best protocol and transmission power to use. For that a regression model is created and trained, to understand how it fits the methodology and obtains the best accuracy. It also compares an edge and cloud computing approach, to check if the decision done directly on the edge node and in real time, without the need of sending any message or flooding the network, could save energy and time in the decision process.

4.3.1. Methodology

The presented methodology aims to create an autonomous solution, capable of selecting the best communication protocol and its transmission power configurations for a smart node, based on its location, the nearby gateways and geography (urban or rural areas, obstacles and distance to the gateway), supported by Machine Learning algorithms that can run directly on the node or with cloud communication. Figure 4.9 shows the system methodology.

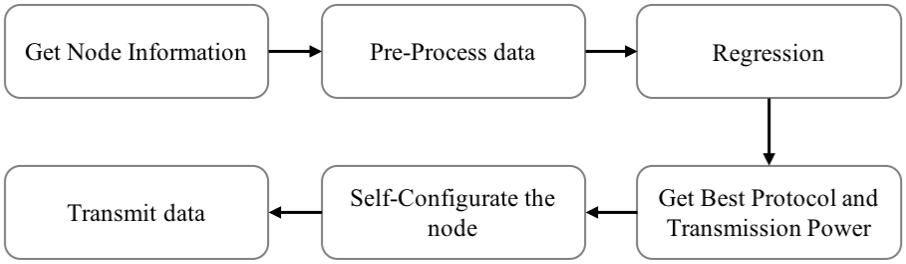


FIGURE 4.9. Point-to-Point Configuration Methodology

As it is possible to see, the methodology is divided into six steps. It starts with the node gathering its location based on GPS coordinates, and pre-process that data. After that, the learning algorithm makes a regression to predict the energy consumption and quality of the link for each of the available protocols and transmission power. Those predictions are then analyzed, and the best transmission power is selected, being the node configured, in an autonomous way, with that value. After that, the node is ready to send messages. From time to time, that can vary based on implementation scenario but that should be done at least once a day, this process is repeated, to ensure the node is always working in the best conditions possible.

Regarding the data processing and regression algorithm, Figure 4.10 shows the detailed process, from the data input to the output of the best protocol and transmission power value.

As described, it receives the node location and starts by comparing it to the list of available gateways, calculating, for each one, a position (X,Y) facing the gateway in the center of a grid (0,0), the distance to the gateway and also any information about possible obstacles in the line of sight. With that information, it creates an array of data that will be used to predict the Received Signal Strength Indicator (RSSI) and energy consumption of the link while using those inputs for each of the communication protocols

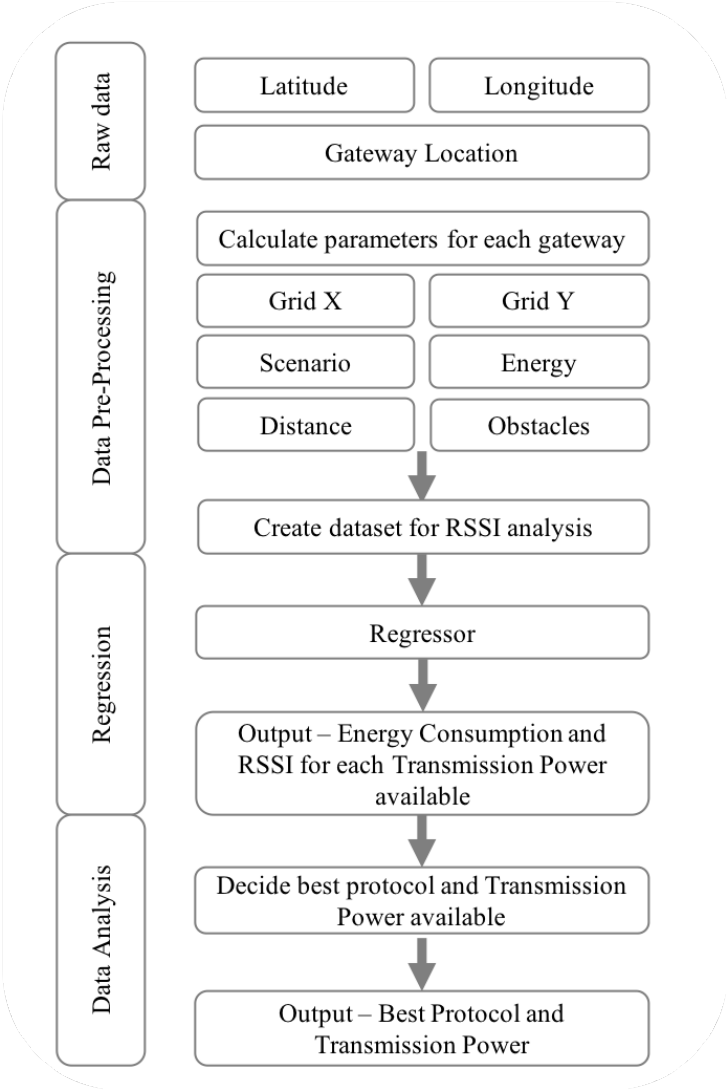


FIGURE 4.10. Point-to-Point Decision Methodology

and transmission power values possible. This is done using a regression model. After the regression, using the output values, three decision models are used to evaluate the best protocol and transmission power to use.

(1) Best Link Model (BLM) - The transmission power is chosen solely based on the best link achieved, i.e., the one that gets the higher RSSI value. With this model, the link will always get the perfect conditions to ensure reliability, not considering the energy consumption. This mode can be used in nodes where information needs to be always delivered in real time, ensuring maximum reliability.

(2) Energy Efficiency Model (EFM) - The transmission power is chosen based on the lowest energy used by a transmission power value capable of sustaining a communication link, even if the RSSI is higher or close to the threshold of the

sensitivity on which each communication protocol can no longer transmit information, putting aside the reliability of the signal, to favor energy efficiency. This mode can be used in nodes where data is not sensitive and crucial, and if some packages are lost, it does not affect the system.

(3) Reliable Link Model (RLM) - The transmission power is chosen based on the lowest energy used by a transmission power value capable of achieving a good communication link, i.e., with a RSSI close to -20dBm of the sensibility threshold, even if when using higher transmission power, a better RSSI values can be achieved. This mode is a middle solution between the previous two, it compromises some of the energy efficiency to guarantee a better connection.

4.3.2. Communication Protocol Configuration Output

As studied in Chapter 3, 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 the data collected in the Point-to-Point protocol comparison done in Section 4.2. The data contains the RSSI and power consumption values of the various protocol's transmissions performed by an IoT smart node in different scenarios around a single gateway, while varying the transmission power value, as well as the distance to the gateway and the number of obstacles in the line of sight.

The dataset is composed of 18448 entries, with the following parameters:

- X – X grid position of the node facing the gateway at position (0,0);
- Y – Y grid position of the node facing the gateway at position (0,0);
- scenario – Characteristics of the transmission scenario (indoor, outdoor, ...);
- distance – Distance, in meters and in line of sight, from the node to the gateway;
- obstacles – Number of obstacles, in line of sight, between the node and the gateway;
- protocol – Communication protocol used for the transmission;
- power – Transmission Power value used for the transmission;
- energy – Energy used to perform the transmission;
- rssi – RSSI value registered from the transmission;

4.3.2.1. *Regression Results.* The training methodology presented in Section 3.2.1 was followed in order to obtain the best Random Forest regression model possible to predict

the energy consumption and RSSI value of the transmissions, based on node position, communication protocol and the transmission power used.

For the energy consumption prediction, following the methodology, the hyper parametrization tuning, showed that the best configuration for the Random Forest regression model was with the following parameters:

- n_estimators – 127
- max_features – 'sqrt'
- max_depth – 70
- min_samples_split – 5
- min_samples_leaf – 1
- bootstrap – False

This model achieved a MAE of 1.503 mA and an accuracy of 99.88%. Figure 4.11 shows the predicted values facing the real values, obtained by the model.

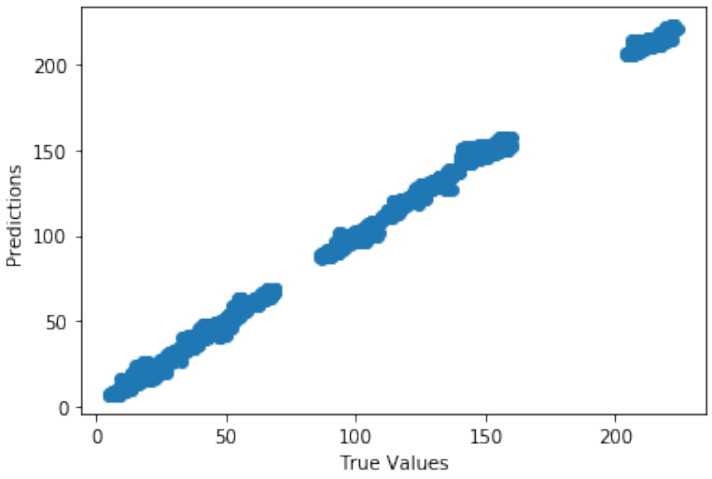


FIGURE 4.11. Point-to-Point Energy Regression Predicted vs Real Values

This shows that the model can predict the energy consumption of a transmission with a 1.503 mA margin of error, which is an acceptable value. Also, as Figure 4.11 shows, the predicted values follow a proportional line, meaning that the model is well fitted for the dataset.

To further validate the model accuracy, and following the methodology presented, the model was validated with a Stratified K-Fold Cross Validation, using 5 folds and 20% of the data as validation points. Figure 4.12 shows the learning curve for the validation test.

It is possible to see, at the end of the learning curve, that both training and validation converge to a similar and lower MAE value, showing that the model is well fitted. The

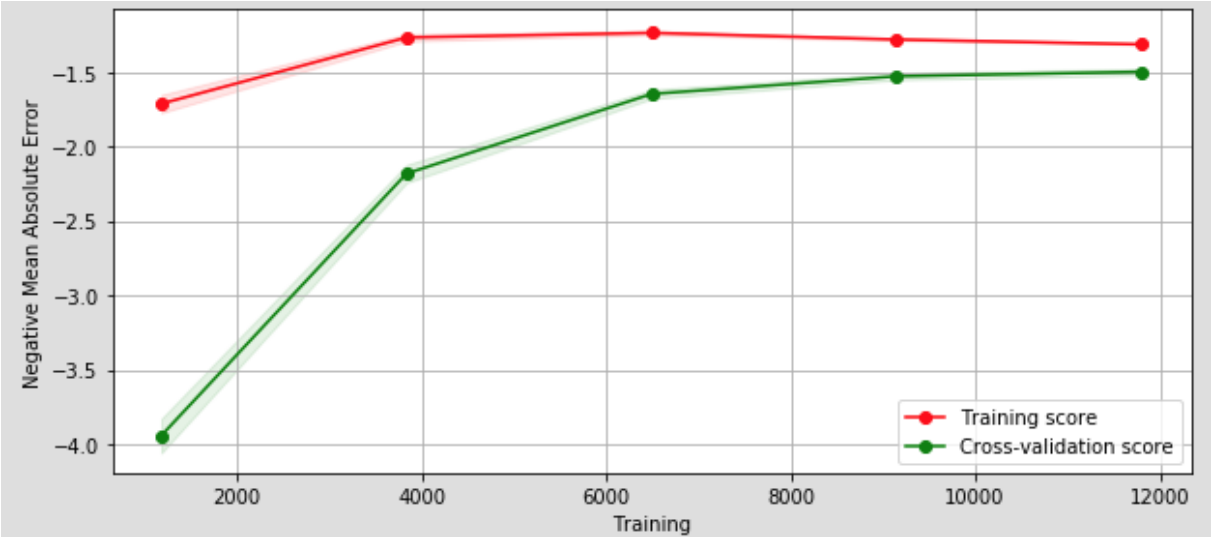


FIGURE 4.12. Point-to-Point Energy Regression Learning Curve

validation MAE was 1.519 mA, 0.015 mA higher than the training MAE, with an accuracy of 99.60%, 0.28% lower, being these values too small to be considered.

As such, is it possible to conclude that the trained model is well fitted and capable of predicting the energy consumption values of communication transmissions, based on location, distance and obstacles to the gateway, and the transmission power value, and can be ported to the node implementation.

Following the training methodology for the RSSI model, the hyper parametrization tuning, showed that the best configuration for the Random Forest regression model was with the following parameters:

- n_estimators – 157
- max_features – 'auto'
- max_depth – 90
- min_samples_split – 5
- min_samples_leaf – 1
- bootstrap – True

This model achieved a MAE of 1.9558 dBm and an accuracy of 98.68%. Figure 4.13 shows the predicted values facing the real values, obtained by the model.

This shows that the model can predict the RSSI of a transmission with a 1.9558 dBm margin of error, which is an acceptable value. Also, as Figure 4.13 shows, the predicted values follow a proportional line, meaning that the model is well fitted for the dataset.

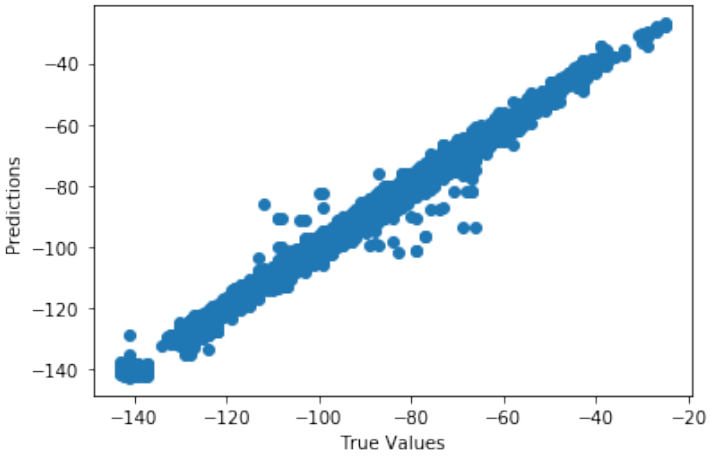


FIGURE 4.13. Point-to-Point RSSI Regression Predicted vs Real Values

To further validate the model accuracy, and following the methodology presented, the model was validated with a Stratified K-Fold Cross Validation, using 5 folds and 20% of the data as validation points. Figure 4.14 shows the learning curve for the validation test.

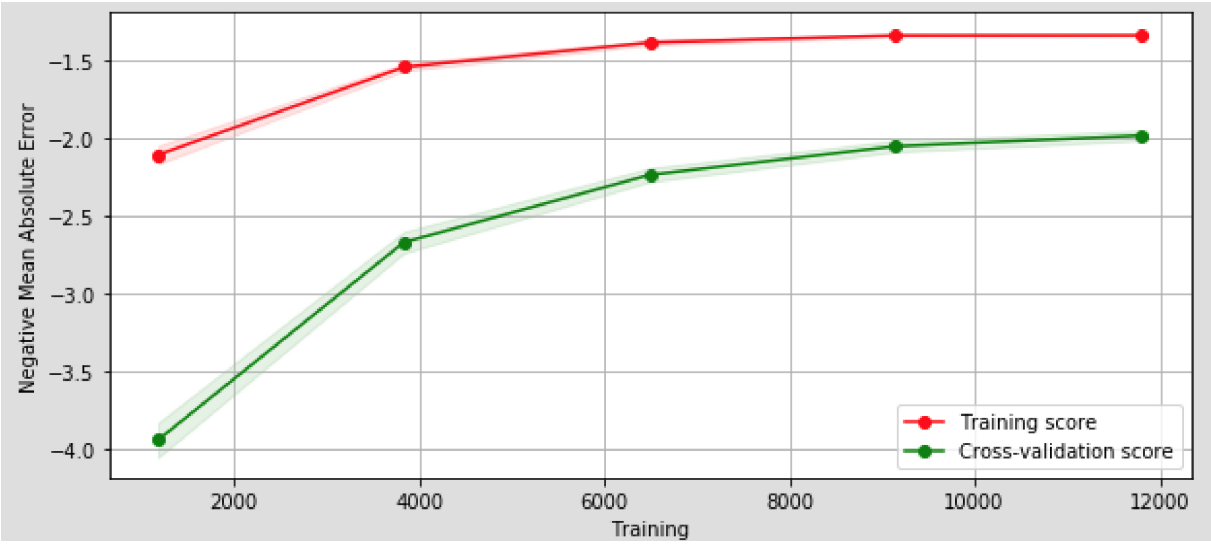


FIGURE 4.14. Point-to-Point RSSI Regression Learning Curve

It is possible to check, at the end of the learning curve, that both training and validation converge to a similar and lower MAE values, showing that the model is well fitted. The validation MAE was 2.031 dBm, 0.07 dBm higher than the training MAE, with an accuracy of 98.42%, 0.25% lower, being these values too small to be considered.

As such, is it possible to conclude that the trained model is well fitted and capable of predicting the RSSI values of communication transmissions, based on location, distance and obstacles to the gateway, and the transmission power value, and can be ported to the node implementation.

4.3.3. Comparing an Edge vs Cloud Approach

To compare if the autonomous configuration system works better using an edge or cloud computing approach, the presented methodology was implemented in a smart node, Figure 4.15, capable of transmitting with all the communication protocols studied.

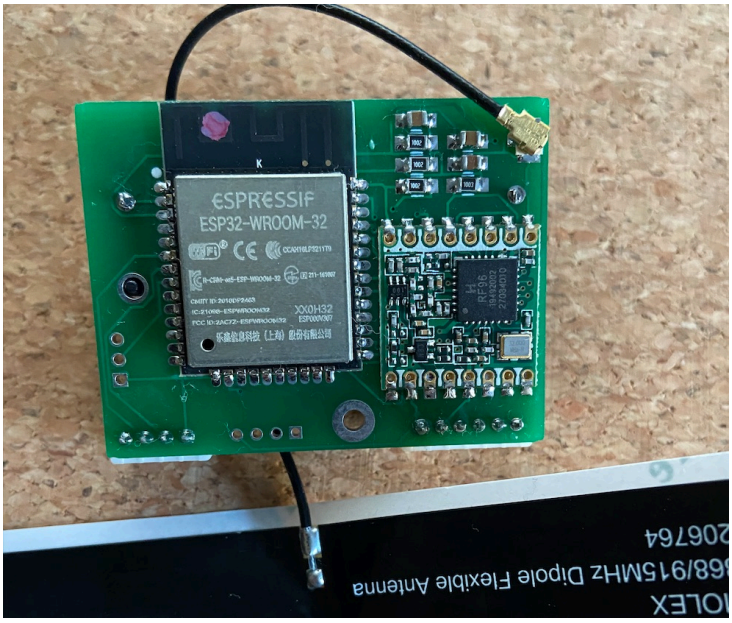


FIGURE 4.15. Smart Node

Several nodes were deployed around one gateway, using the same urban environment as the one used on Section 4.2.2.4, covering an implementation area of 36ha, with a radius of 600 meters from the gateway position. The nodes were deployed for a period of one and a half months, fifteen days using the BLM scenario, fifteen days using the EML scenario and the other fifteen using the RLM scenario, being the ones working on the edge computing model self-configured every 48 hours, or if any message was not able to be delivered, since these do not need external messages and require less energy, and the ones working with the cloud computing model self-configured once a week, since they need to transmit a message via the gateway to the cloud asking for the configuration parameters, therefore needing more power to perform this task.

Before the implementation, the regression model was ported to an edge computing model capable of running in the smart node. For this, the methodology presented in Section 3.3 was used. Table 4.2 shows the ported model characteristics facing the trained cloud model.

The cloud computing model, with an MAE of 1.503 mA and 1.9558 dBm, originated an 83.2 and 97.8 MB file containing the Random Forest regression model for energy and

TABLE 4.2. Point-to-Point Configuration System Edge Computing Model Characteristics

Feature	Model	MAE [dBm]	File Size [MB]	Estimators	Depth
Energy	Cloud Computing	1.503	83.2	127	70
	Edge Computing	3.203	0.437	15	7
RSSI	Cloud Computing	1.9558	97.8	157	90
	Edge Computing	3.992	0.503	15	7

RSSI prediction, respectively. As said, the microcontrollers are not capable of sustaining files that big, so for the edge computing port, the number of estimators and depth was adjusted, in order to create a smaller file. As studied in Section 3.3, it is possible to create files up to 99.99% smaller in size, when reducing 90% of estimators and depth, with this affecting around 25% the accuracy of a regression. The obtained edge computing model, for each regression, contains 15 estimators, 90% less, and a depth of 7, 92% less, originating models with a 3.203 mA and 3.992 dBm MAE, with file sizes of 437 and 503 kB, 99.5% smaller. Although this model has almost doubled the MAE of the original model, it is still a valid result for an energy and link quality regression, and one capable of running on an edge device.

After one and half months of deployment, running all the scenarios described, the results were analyzed and are outlined using a scatter plot with a linear distribution, showing the average Transmission Power and RSSI used by each node during the implementation period, as well as the chosen protocol, for both edge and cloud models. Table 4.3 summarize the obtained results.

4.3.3.1. ***Best Link Model Scenario.*** As described in the methodology, the Best Link Model (BLM) chooses the protocol and transmission power based solely on the best link achieved, i.e., the one that gets the highest RSSI value.

In this scenario, both edge and cloud computing models choose LoRa as the main protocol to use, in 92% and 93% of the cases, respectively. For edge computing, the remaining cases used Zigbee, 3%, or RF, 5%, as for cloud computing, Zigbee, 4%, and RF, 3%, were the selected ones. These variations in the selection of the communication protocols come in line with LoRa being the best overall protocol, as studied in Section 4.2, for urban environments and long distances, with the other protocols being selected only for close range nodes without obstacles.

TABLE 4.3. Implementation Results

	Model	Protocol	P_{tx} [dBm]	RSSI [dBm]	I_{tx} [mA]
BLM	EC	LoRa (92%)	20 (-)	-115.36 (-)	90 (-)
	CC	LoRa (93%)	20 (-[2])	-117.48 (+2%[2])	112 (+24%[2])
EFM	EC	LoRa (87%)	10 (-50%[1])	-130.25 (+13%[1])	29 (-68%[1])
	CC	LoRa (84%)	11 (-45%[1] || +10%[2])	-123.88 (+5%[1] || -5%[2])	43 (-62%[1] || +48%[2])
RLM	EC	LoRa (98%)	14 (-30%[1])	-124.15 (+8%[1])	50 (-45%[1])
	CC	LoRa (96%)	16 (-20%[1] || +14%[2])	-120.24 (+2%[1] || -3%[2])	68 (-39%[1] || +36%[2])

1 - Comparing to the BLM scenario under the same model

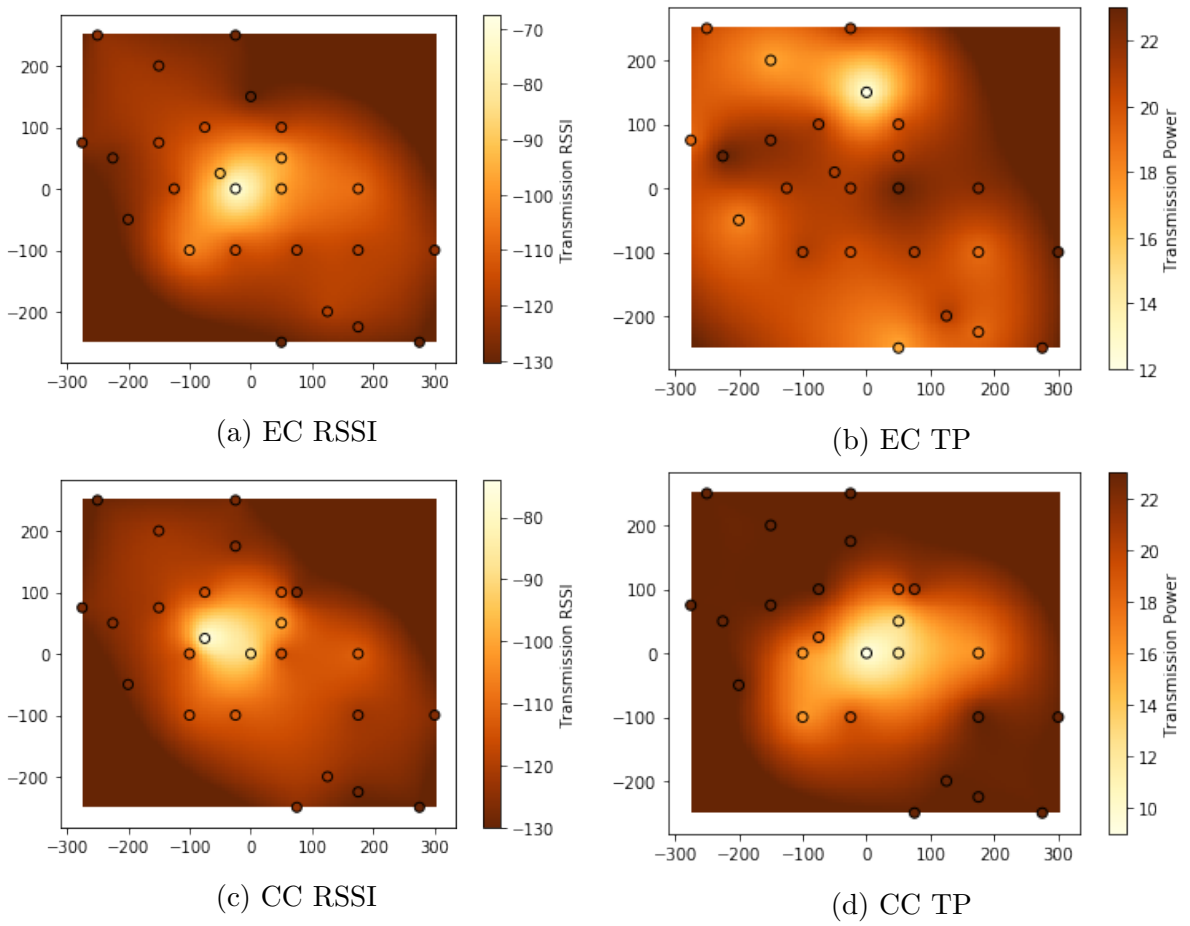
2 - Comparing to the EC model under the same scenario

The LoRa scenarios, being the vast majority of cases, will be the only ones analyzed in this scenario. The results obtained for RSSI and transmission power, for both edge and cloud computing, can be found in Figure 4.16. Table 4.4 shows the average values for this scenario.

TABLE 4.4. BLM Scenario Results

	P_{tx} [dBm]	RSSI [dBm]	I_{tx} [mA]
Edge Computing	20	-115.36	90
Cloud Computing	20	-117.48	112

It is possible to check, Figure 4.16 (a), that when using Edge Computing alongside the BLM mode, the closest nodes transmit with a average value of -85 dBm, while mid-range and further nodes transmit with an average -110 dBm and -125 dBm, respectfully. For the same distance, the average transmission power value, Figure 4.16 (b), is 18, 20 and 22, respectively.



(a) EC RSSI

(b) EC TP

(c) CC RSSI

(d) CC TP

FIGURE 4.16. Results for BLM Scenario

One thing that is interesting to see, is that some mid-range nodes can transmit with an average of 14 dBm of transmission power. Also, the further nodes with more obstacles were not able to create a connection.

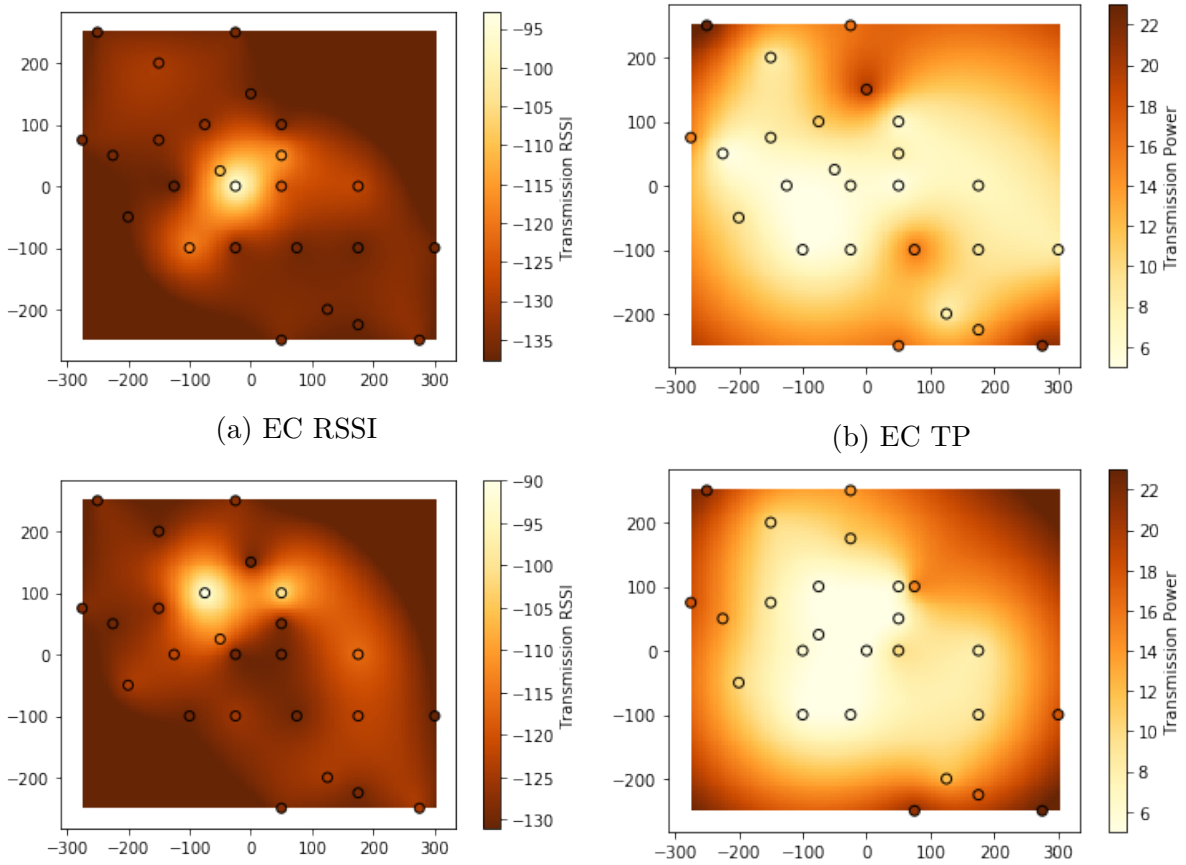
As for the Cloud Computing model, when associated with BLM mode, in terms of RSSI, it has similar results as the Edge Computing model, as can be seen when comparing Figure 4.16 (c) and Figure 4.16 (a), with the connection link being only 2% worst, about -2 dBm. In terms of transmission power, although both models achieve an average of 20 dBm, when looking to Figure 4.16 (d) and Figure 4.16 (b), it is possible to check that in the Cloud Computing model the further nodes have higher transmission power, while the Edge Computing presents a more even distribution between all nodes.

In terms of energy consumption, the nodes using the Edge Computing model used less 24% power facing the Cloud Computing nodes. As such, comparing the two models it is possible to check that the edge computing achieves better results on all fields, using 20 mA less, while increasing the quality of the link connection by 2 dBm. So, a better connection can be achieved with a lower power consumption.

4.3.3.2. *Energy Efficiency Model Scenario.* The Energy Efficiency Model (EFM) chooses the protocol and transmission power based on the lowest energy value of a transmission power that can achieve a connection.

As in the BLM mode, in this scenario, both edge and cloud computing models choose LoRa as the main protocol to use, although in a smaller value, 87% and 84% of the cases, respectively. For edge computing, the remaining cases used Zigbee, 6%, RF, 5%, and BLE, 2%, as for cloud computing, Zigbee, 9%, RF, 5%, and BLE, 2%, were the selected ones. Once again, the variations in the protocol chosen are accounted for only in the nodes in close range, without obstacles.

The results obtained for RSSI and transmission power, for both edge and cloud computing, can be found in Figure 4.17, being once again presented only the LoRa results. Table 4.5 shows the average values for this scenario.



(a) EC RSSI

(b) EC TP

(c) CC RSSI

(d) CC TP

FIGURE 4.17. Results for EFM Scenario

The results show that when using Edge Computing alongside EFM mode, the quality of the signal decreases while lower transmission power values are used, facing the BLM mode. Figure 4.17 (a) shows that the closest nodes transmit with an average value of

TABLE 4.5. EFM Scenario Results

	$\mathbf{P_{tx}}$ [dBm]	RSSI [dBm]	$\mathbf{I_{tx}}$ [mA]
Edge Computing	10	-130.25	29
Cloud Computing	11	-123.88	43

-115 dBm, while mid-range and further nodes transmit with an average -123 dBm and -131 dBm, respectively. For the same distance, Figure 4.17 (b), the average transmission power value is 6, 10 and 17, respectively. As in the previous scenario, the further nodes with more obstacles were not able to create a connection.

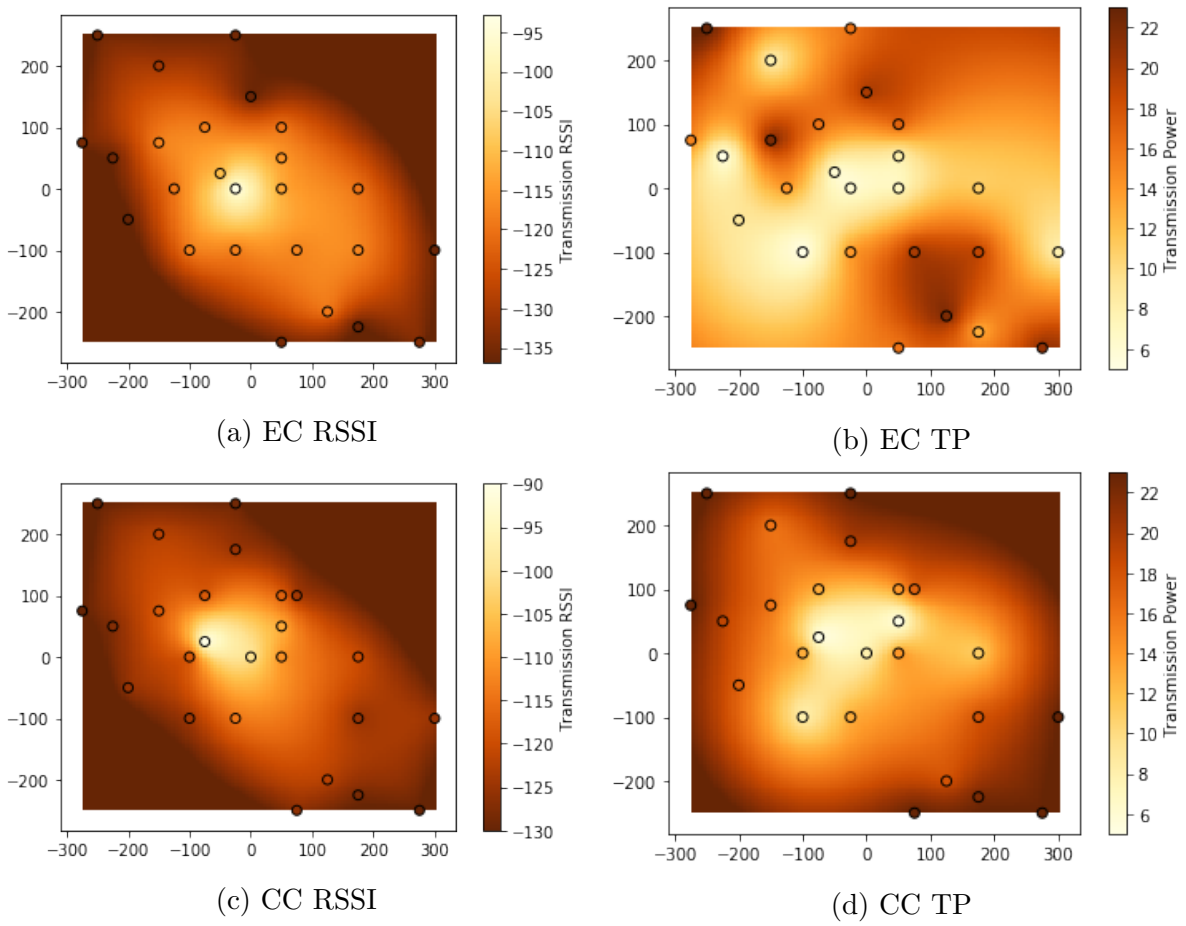
In the Cloud Computing model, as in the BLM scenario, a similar behaviour can be found when comparing with the Edge Computing, Figure 4.17 (c). Contrarily to the BLM scenario, in the EFM mode, this model achieved a better link connection, being 7 dBm higher, or -5%. This is partially caused by the higher transmission power being used by the Cloud Computing model, Figure 4.17 (d), that is, on average, 1 dBm higher than the Edge Computing, allowing for a better signal quality.

As for energy consumption, the Cloud Computing, with a higher transmission power, consumed on average 43mA, which is 48% more than the Edge Computing model. Considering these results, the Edge Computing model, besides the 5% lower link quality, presents once again the best alternative, mainly when considering this is an energy efficient mode, focused only on decreasing the power consumption. So, a worst connection can be achieved, but is it done with a lower power consumption.

4.3.3.3. *Reliable Link Model Scenario.* Finally, the Reliable Link Model (RLM) compromised some of the energy efficiency of the ELM model and the link quality of the BLM model, choosing the protocol and transmission power based on the lowest energy value of a transmission power that can achieve a good connection, i.e., close to -20 dBm of the sensibility threshold for that network.

Following the path of the previous modes, in this scenario, both edge and cloud computing models choose LoRa as the main protocol to use, with a higher value, 98% and 96% of the cases, respectively. For edge computing, the remaining cases used Zigbee, 2%, as for cloud computing, Zigbee, 3%, and RF, 1%, were the selected ones.

As for the previous scenarios, only the LoRa results will be analysed, with the obtained for RSSI and transmission power results, for both edge and cloud computing, presented in Figure 4.18. Table 4.6 shows the average values for this scenario.



(a) EC RSSI

(b) EC TP

(c) CC RSSI

(d) CC TP

FIGURE 4.18. Results for RLM Scenario

TABLE 4.6. RLM Scenario Results

	$\mathbf{P_{tx}}$ [dBm]	RSSI [dBm]	$\mathbf{I_{tx}}$ [mA]
Edge Computing	14	-124.15	50
Cloud Computing	16	-120.24	68

When using the Edge Computing model with the RLM mode, as in the EFM mode, the quality of the signal decreases while lower transmission power values are used, facing the BLM mode, but achieves a better quality of signal with slightly higher transmission power values, facing the EFM mode. The closest nodes, Figure 4.18 (a), transmit with an average value of -105 dBm, while mid-range and further nodes transmit with an average -115 dBm and -128 dBm, respectively. For the same distance, Figure 4.18 (b), the average transmission power value is 7, 12 and 18, respectively.

As in BLM mode, some mid-range nodes can transmit with a higher transmission power value, with an average of 22 dBm. As in the previous scenarios, the further nodes with more obstacles were not able to create a connection.

Regarding the Cloud Computing model, as in the previous modes, it follows the Edge Computing in terms of behaviour, Figure 4.18 (c), having a better communication link, 3% higher, than the Edge Computing model. This, as in the EFM mode, can be justified by the use of a higher transmission power, Figure 4.18 (d), in this case by 2 dBm. Also, when comparing the transmission power results, it is possible to check that, as in the BLM mode, the further nodes have a higher transmission power than the Edge Computing nodes that have a more even distribution.

This higher transmission power, as in the EFM mode, draws more power from the nodes, consuming 36% more than the Edge Computing nodes. Considering this and the slight difference between link quality, of only 3%, the Edge Computing gets, once again, the advantage facing the Cloud Computing. So, a slightly worse connection can be achieved, but is it done with a lower power consumption.

4.3.4. Discussion

This section presented a methodology for an autonomous implementation of a self-configuring smart node supported by Machine Learning, that uses Random Forest regressions to predict the energy consumption and link quality of a connection and then chooses the best Transmission Power and communication protocol to use.

The first thing to conclude is that with proper configuration, the smart node can act in a lower power fashion, creating more sustainable networks.

Regarding the computation model, capable of predicting the energy consumption and RSSI of a wireless communication, based on the location of the node, distance and obstacles to the gateway and the transmission power value, Random Forest achieved an accuracy of 99.88% and 98.68%, with a margin of error of 1.504 mA and 1.9558 dBm, respectively for energy and RSSI prediction. This proves to be an efficient way to configure the node without human intervention, with the nodes being self-configured every 48 hours, or if any message was not able to be delivered.

One of the other goals of this research was to understand how the edge computing methodology faces a cloud computing methodology for deciding which is the best protocol and transmission power value for a smart node to transmit messages. The edge computing methodology can achieve better results, while sometimes having a lower quality of service, although only by a slight margin, proving to be a better solution, since it takes less time to decide and configure the node, being done locally and without external inputs.

Finally, three modes of selecting the best transmission power value were presented, each with a specific role depending on the needs of the network. The BLM mode proves to be reliable, with the best quality link being achieved, with RSSI values higher, but with more energy being used. The EFL mode proves to be a solution for low power nodes, using 68% less energy but compromising the reliability of the network by 13%. The RLM mode is a balanced solution, it can save 45% more energy than the BLM mode with a 7% better quality than the EFL mode.

By applying this methodology to a network, not only can it extend the life cycle of the nodes but also reduce the need for maintenance and interference between nodes, creating a more sustainable and reliable network.

4.4. Cloud Communication Configuration System

Cloud communication is how an IoT system is capable of sending the gathered data and receiving tasks or interacting with the users. As described, multiple techniques are available for these exchanges of messages, each of them with its advantages and disadvantages. As LPWAN are increasingly available for the general public, systems no longer have to rely only on WiFi and cellular networks to connect to the cloud services. Nevertheless, and as each scenario can have different coverage from multiple protocols, and with each having different requirements, such as price, energy consumption or data package size, the use of the correct network protocol can improve the efficiency of an IoT system.

As for the Point-to-Point communication system, this section presents a methodology of an autonomous configuration system for cloud communications in smart nodes supported by Machine Learning, that uses regressions to predict the availability of protocols and the energy consumption and link quality of those connections, in order to choose the best protocol for transmission. Besides the methodology and the model for the regression techniques, it will also present a comparison between an edge and cloud computing approach to evaluate the latency and energy consumption of the nodes to reach a decision.

4.4.1. Methodology

The presented methodology aims to create an autonomous solution, capable of selecting the best communication protocol for cloud communication in a smart node, based on its location, the available networks covering that area and its base stations, and the price and energy consumption of transmitting a message, supported by Machine Learning

algorithms that can run directly on the nodes or with cloud communication. Figure 4.19 shows the system methodology.

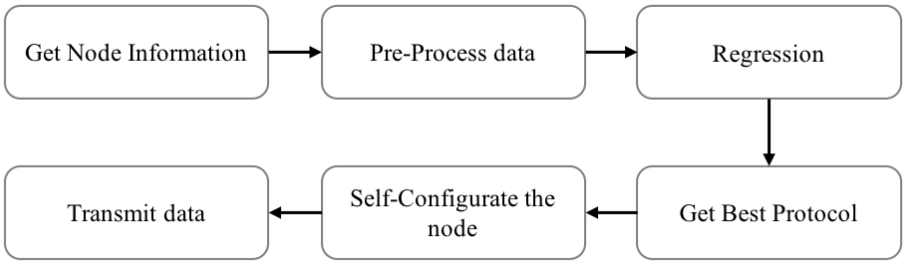


FIGURE 4.19. Cloud Configuration Methodology

As it is possible to see, the methodology is divided into six steps. It starts with the node gathering its location based on GPS coordinates, and pre-process that data. After that, the learning algorithm makes a regression to predict the quality of the link for each of the available protocols. Those predictions are then analyzed, and the best available protocol is selected, being the node configured, in an autonomous way. After that, the node is ready to send messages. Every 12 hours, or if any message was not able to be delivered, this process is repeated, to ensure the node is always working in the best conditions possible.

Regarding the data processing and regression algorithm, Figure 4.20 shows the detailed process, from the data input to the output of the best protocol and configuration value.

The configuration system receives the node location and starts by comparing to the list of available base stations for each protocol, calculating, based on each protocol range, what are the ones in range of the node location and the distance between the node and the base station. Then, using the available base station in range, it creates an array of data that will be used to predict the RSSI of the communication link using the associated protocol to each of the base stations, through a regression model. After the regression, using the output values, the decision for the protocol to use is done using a score for each of the available protocols. Table 4.7 shows how the scoring is done.

Each category has the same importance and the amount of points awarded for each category, based on the protocol, follows the threshold values found in Table 4.8.

For each decision, the available protocols are scored and the highest score is the one used for the communication transmission. This way is possible to choose the best configuration based on availability, price and energy consumption, creating a methodology that adapts to the node location and network availability, supports sustainability while

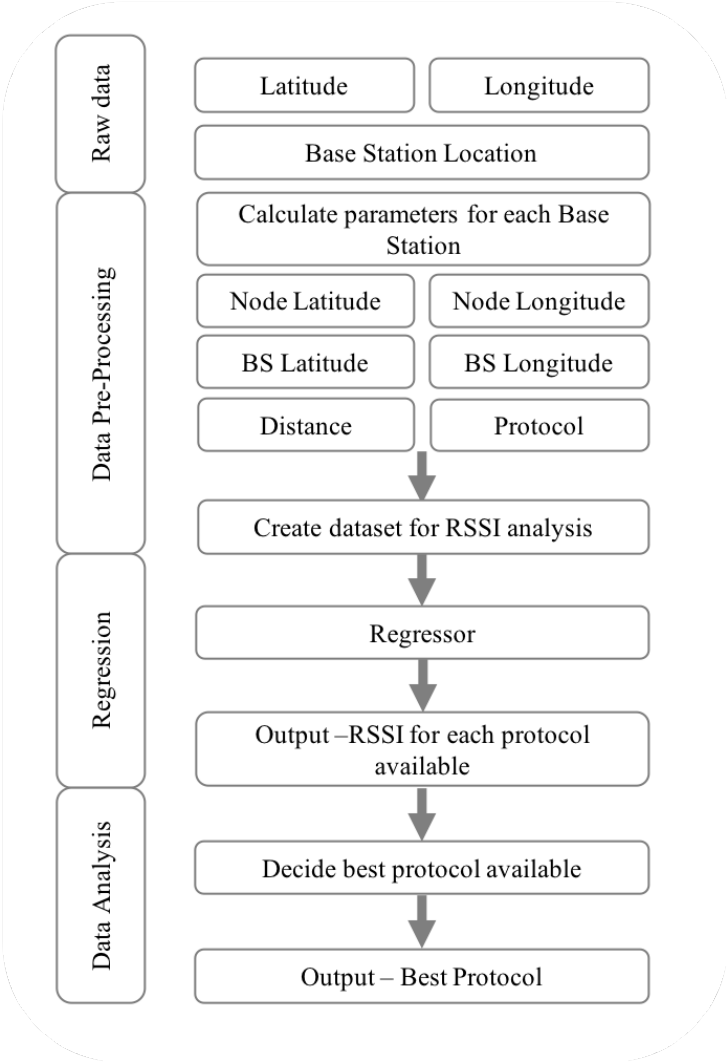


FIGURE 4.20. Cloud Decision Methodology

reducing the costs for the user and improving the system reliability and efficiency. It is also possible to change the scoring system to favour any specific parameter, e.g., if the goal is to transmit always using the lower cost or power consumption.

4.4.2. Communication Protocol Configuration Output

As studied in Chapter 3, 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 [**88**], "a European Commission crowd-sourcing project designed to gather and share radio spectrum data about mobile telephony coverage, WIFI 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

TABLE 4.7. Cloud Configuration Scoring Description

Category	**Points Available**	**Evaluation Process**
Message Size	0	Message size is larger than the maximum package size
	1	Message is smaller than the maximum package size
Price	6	Free
	5 to 1	In ascending order of price
Message Limit	0	Communication is limited to any amount of messages
	1	Communication is unlimited
Energy Consumption	6 to 1	In ascending order of energy consumption
Average RSSI Predicted	6 to 1	In descending order of RSSI

TABLE 4.8. Cloud Configuration Protocol Thresholds

Category	**Wi-Fi**	**2G**	**3G**	**4G**	**LoRaWAN**	**SigFox**
Message Size [bytes]	-	-	-	-	51	12
Price	-	0.10€ per MB	0.10€ per MB	0.10€ per MB	-	16.13€ per year
Message Limit	-	-	-	-	1 per minute	140 per day
Energy Consumption [mA]	120	250	150	100	70	70

getting on their phone – WIFI, 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 [**112**], 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 (BS)".

To complement these data, the information from the possible receiving BS for each entry was added. For LoRaWAN, the dataset already includes the associated BS and its location; for SigFox, the dataset only provides the BS id, but not the location, so using all the points received by each BS, the location was estimated; for the cellular protocols (2G, 3G and 4G) the OpenCelliD database, the largest open-data of cell towers [**113**],

was used to associate each entry with the cells in range; and finally, for the WiFi data the Radiocell database was used, containing information about 10 million WiFi base stations [**114**], once again, to associate each entry with the cells in range.

As each of these databases have different parameters, units and configuration, some data pre-processing was needed to join all the information into one dataset. This included removing unnecessary data or data only available on one dataset, identifying each protocol with its ID and adding the range of each network.

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 1349428 entries, with the following parameters:

- Latitude – Latitude of the node;
- Longitude – Longitude of the node;
- Protocol – Protocol used in the data transmission: 1 - Wi-Fi, 2 - 2G, 3 - 3G, 4 - 4G, 5 - LoRaWAN, 6 - SigFox;
- BS_Code – Base station identifier;
- BS_Lat – Latitude of the Base station;
- BS_Long – Longitude of the Base station;
- BS_Distance – Distance between the node and the Base station;
- RSSI – Link quality of the data transmission;

To follow the presented methodology, another dataset was needed, containing only the location of each Base Station. Using the previously described data, it was possible to create a dataset composed of 27682 entries, with the following parameters:

- BS_Network – Protocol used in the Base Station: 1 - Wi-Fi, 2 - 2G, 3 - 3G, 4 - 4G, 5 - LoRaWAN, 6 - SigFox;
- BS_Lat – Latitude of the Base station;
- BS_Long – Longitude of the Base station;
- BS_Code – Base station identifier;
- BS_Range – Base station range;

4.4.2.1. ***Regression Results.*** Once again, the training methodology presented in Section 3.2.1 was followed in order to obtain the best Random Forest regression model possible

to predict the RSSI value of the transmissions, based on node position, communication protocol, nearby Base Stations and its distance.

Following the methodology, the hyper parametrization tuning, showed that the best configuration for the Random Forest regression model was with the following parameters:

- n_estimators – 158
- max_features – 'auto'
- max_depth – 123
- min_samples_split – 5
- min_samples_leaf – 1
- bootstrap – False

This model achieved a MAE of 2.616 dBm and an accuracy of 96.82%. Figure 4.21 shows the predicted values facing the real values, obtained by the model.

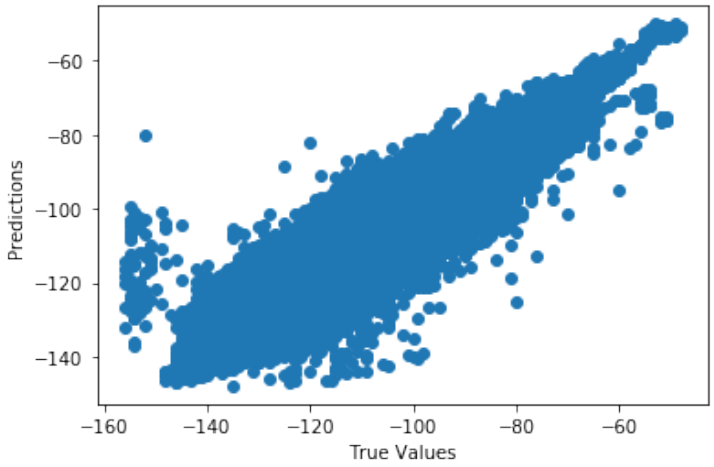


FIGURE 4.21. Cloud RSSI Regression Predicted vs Real Values

This shows that the model can predict the RSSI of a transmission with a 2.6168 margin of error, which is an acceptable value.

To further validate the model accuracy, and following the methodology presented, the model was validated with a Stratified K-Fold Cross Validation, using 5 folds and 20% of the data as validation points. Figure 4.22 shows the learning curve for the validation test.

It is possible to see, at the end of the learning curve, that both training and validation converge to a similar and lower MAE value, showing that the model is well fitted. The validation MAE was 2.618 dBm, 0.002 dBm higher than the training MAE, with an accuracy of 96.27%, 0.56% lower, being these values too small to be considered.

As such, is it possible to conclude that the trained model is well fitted and capable of predicting the RSSI values of communication transmissions, based on location, type

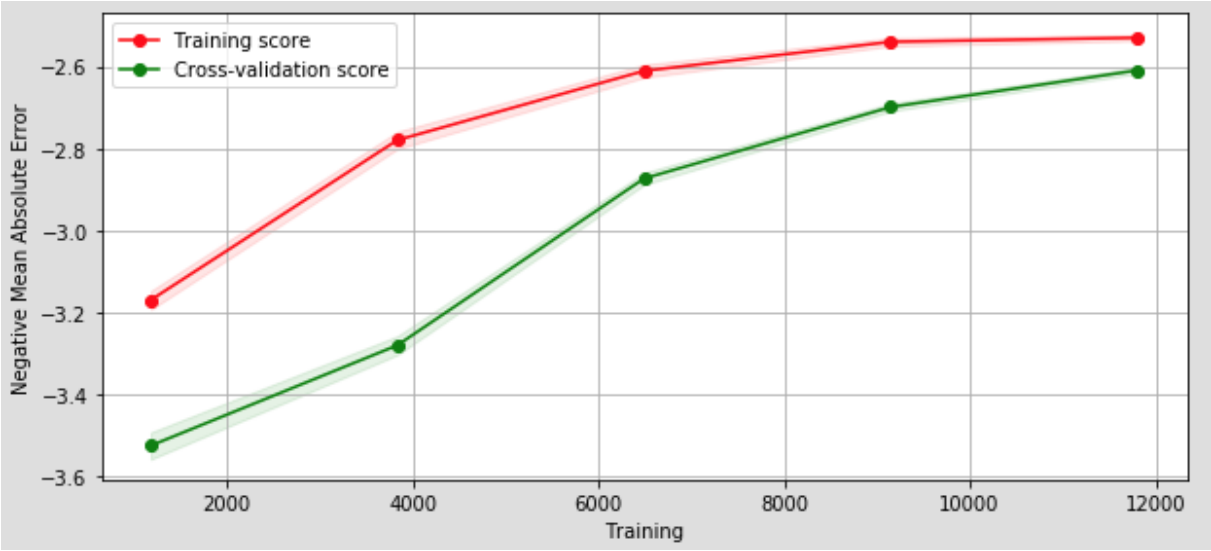


FIGURE 4.22. Cloud RSSI Regression Learning Curve

of protocol, the nearby Base Stations and its distance, and can be ported to the node implementation.

4.4.3. Edge vs Cloud Approach Results

To validate the autonomous configuration system for cloud communications and compare if it works better using an edge or cloud computing approach, the presented methodology and regression model was implemented in a smart node.

Since the data used to train the model was from Antwerp, it was impossible to perform an in site deployment, as in the Point-to-Point configuration system, so a simulation was performed. The smart node used in the previous test, Figure 4.15, was configured to use the proposed methodology, while simulating its position, to use Antwerp coordinates, and the protocol to use.

Before the simulation the regression model was ported to an edge computing model capable of running in the smart node. For this, the methodology presented in Section 3.3 was used. Table 4.9 shows the ported model characteristics facing the trained cloud model.

TABLE 4.9. Cloud Configuration System Edge Computing Model Characteristics

Model	MAE [dBm]	File Size [MB]	Estimators	Depth
Cloud Computing	2.6168	5126	158	123
Edge Computing	4.3831	1.723	25	8

The cloud computing model, with an MAE of 2.6168 dBm, originated an 5.1 GB file containing the Random Forest regression model with 158 estimators and a depth of 123.

As said, the microcontrollers are not capable of sustaining a file that big, so for the edge computing port, the number of estimators and depth was adjusted, in order to create a smaller file. As studied in Section 3.3, it is possible to create files up to 99.99% smaller in size, when reducing 90% of estimators and depth, with this affecting around 25% the accuracy of a regression. The obtained edge computing model contains 25 estimators, 85% less, and a depth of 8, 94% less, originating a model with a 4.3231 dBm MAE, 1.7063 dBm higher, with a file size of 1.732 MB, 96.63% smaller. Although this model has almost doubled the MAE of the original model, it is still a valid result for a link quality regression, and one capable of running on an edge device.

To follow the presented methodology, for the edge computing approach, besides the model C file, also the dataset CSV file, containing the base stations information, needed to be uploaded to the node. This proved to be impossible, since this last file has 2 MB of size, and in conjunction with the model file and the application file surpass the 4 MB Flash limit of the node. From a methodology point of view, this is a major disadvantage for the edge computing approach, since the node and the model are intended to work anywhere in the world, and if it can not cope with the data for only one city, it will not work in a larger deployment.

To surpass this limitation, the methodology for the edge computing approach was revised to remove the need of the base station knowledge, being the prediction of link quality solely made based on node location and protocol. This new methodology assumes that all networks are in range and uses only the training data, without the associated base station, to understand if a connection is possible.

For this, a new model was trained, following the same methodology of training presented in Section 3.2.1. This revised model achieved a MAE of 5.482 dBm and an accuracy of 87.93%. Although this is 2.8 dBm higher than the previous model, it can still be considered acceptable for a link communication, since it is under 9 dBm [**70**]. When ported to an edge computing model, the MAE increases to 6.1298 dBm.

Besides this new model, and to be able to properly compare the edge and cloud computing approaches using the original methodology, the base station file was shortened by 70%, to be able to fit in the microcontroller and perform the original methodology. As such, three models will be tested and compared. Table 4.10 shows the models information.

The simulation was performed on 500 randomly selected points in the city of Antwerp, using both edge and cloud computing approaches. For each simulation the available

TABLE 4.10. Revised Cloud Configuration System Edge Computing Model Characteristics

Model	MAE [dBm]	File Size [MB]	Estimators	Depth
Cloud Computing	2.6168	5126	158	123
Edge Computing	4.3831	1.732	25	8
Revised Edge Computing	6.1298	1.918	25	8

protocol and the number of gateways in range, the awarded points, the selected protocol and the estimated RSSI were recorded. Table 4.11 shows the obtained results.

TABLE 4.11. Cloud Configuration System Simulation Results

Model	Protocols				
	ID	Available	Range	Avg. Points	Selected
Cloud	Wi-Fi	74	103	9.80	8
	2G	258	9	7.76	26
	3G	313	13	8.61	21
	4G	281	8	9.60	75
	LoRaWAN	306	3	10.31	294
	SigFox	155	13	9.30	15
	None	-	-	-	61
Edge	Wi-Fi	49	42	9.93	5
	2G	138	4	8.02	19
	3G	208	5	8.98	18
	4G	186	3	9.81	53
	LoRaWAN	216	2	10.40	192
	SigFox	86	7	9.73	12
	None	-	-	-	201
Revised Edge	Wi-Fi	-	-	9.37	104
	2G	-	-	5.19	4
	3G	-	-	6.23	6
	4G	-	-	8.68	14
	LoRaWAN	-	-	9.81	202
	SigFox	-	-	9.49	170
	None	-	-	-	-

For the cloud computing model, LoRaWAN was the most selected protocol, with almost 59% of the locations. The other protocols were chosen between 4 and 13% each.

Also, in 61 locations, none of the protocols were chosen. In terms of the points decision system, it is possible to see that there is an even distribution of points among the protocols, which can be directly related to the availability of the networks on each point, with none being available on all the performed test, meaning that the lowest scoring protocols were benefited from the unavailability of the higher scoring networks, achieving better results. Wi-Fi was the least selected protocol, which was a surprise, since it is a free protocol, but it can be justified by the low availability, only 74 times. Overall, this model proved to be a good choice for the decision process of the cloud communication system.

For the original edge computing model, once again LoRaWAN was the most selected model, in 38% of the locations, with the other protocols ranging from 1 to 10%. In this scenario 201 locations had no protocol selected, which is over 3 times more than in the cloud computing scenario. This is justified by the lack of base stations available to the model to analyse, as only 30% of the original base station database was given to the edge model. This impact is also visible in the number of available base stations for each protocol, which is on average 60% lower than the cloud computing scenario. As fewer protocols are available, it also affects the point system, with an even lower deviation on the results. As was already possible to conclude, this model is not a good choice for the decision process, not only since it can not support the full methodology, but it also provides worse results.

To cope with this failed edge computing methodology, a revised edge model was tested. In this case, as the base station knowledge was eliminated from the methodology, the model assumed that there was always availability from all networks. That methodology created an unreal decision system since, as proven by the cloud computing model, where multiple locations did not have some protocols available. That can be proven by the awarded point for each network, as the lowest scoring protocols always score low, whereas the higher scoring protocols always score high and are the chosen ones. Although, in this scenario there is no protocol that stands from the other, as LoRaWAN, SigFox and Wi-Fi were the selected ones, on average, in 30% of the locations each. Beside having its disadvantages, this model can be used when a cloud computing solution is not available.

Beside the presented results, to further evaluate the differences between edge and cloud computing for the cloud communication configuration system, the average decision time, energy consumption and number of messages exchanged were also recorded. Table 4.12 shows the obtained results.

TABLE 4.12. Analysis Time & Power Consumption Results

Model	MAE [dBm]	Average Time [μs]	Average Energy [mA]	Average Messages Exchanged
Cloud Computing	2.6168	681610	118	2
Edge Computing	4.3831	3532849	83	0
Revised Edge Computing	6.1298	1197	52	0

Although the original edge computing model was already discarded, this decision time analysis allows to prove, even further, the unacceptability of that model, as it takes over 3.5 seconds to make a decision, almost 5 times more than the cloud computing model. This latency between the request and the decision can be justified by the lack of computation power of the edge node, which takes time to open the base station file and go through every single line to find if it is in range of the device. As for the revised edge computing model, since the base station knowledge was discarded, only 1197 μs are needed to reach a decision, although, as already stated, this scenario presents its own problems versus the cloud computing model.

In terms of energy used, as expected, the cloud computing model requires more energy, since it needs to exchange messages with the cloud servers, requiring 118 mA, 56% more than the revised edge computing model, with only 52 mA.

4.4.4. Discussion

This section presented a methodology for an autonomous implementation of a self-configuring cloud communication system supported by Machine Learning, that uses Random Forest regressions to predict the link quality of a connection and then chooses the best communication protocol to use.

Regarding the computation model, capable of predicting the RSSI of a wireless communication, based on the location of the node, protocol, and nearby base stations and its distance, Random Forest achieved an accuracy of 96.82%, with a margin of error of 2.6168 dBm. This proves to be an efficient way to configure the system without human intervention.

One of the other goals of this research was to understand how the edge computing methodology faces a cloud computing methodology for deciding which is the best protocol to transmit messages. The edge computing methodology proved to not be capable of handling the defined methodology, as the edge nodes does not provide the needed computation power nor flash memory capacity to have all the necessary knowledge to follow the

methodology. Even with a revised edge computing model, that does not depend on third party knowledge, edge computing was not able to perform at the level of cloud computing. As such, although with a higher decision time and power consumption, cloud computing achieved the best results, with a MAE 4 dBm lower.

On a final note, to implement the cloud computing methodology outside of the simulation environment, it is necessary to define an initial communication protocol to use for sending and receiving the decision request and result, as these messages need to be exchanged with the servers prior to the node self-configuration.

4.5. Remarks

This chapter establishes the basis for the importance of the autonomous communications systems in the developed solution. From the introduction of the concepts to its research and implementation, it presented a detailed approach of creating and evaluating the autonomous configuration of the communication systems and how its implementation will create a more sustainable and reliable solution.

With the goal of assessing the best communication protocol for point-to-point communication inside a WSN, several were studied, among them ESPNow, Bluetooth Low Energy, FSK Radio Frequency, LoRa and ZigBee. All these protocols were tested under different scenarios and configurations, being possible to conclude that, although with a higher power consumption needed to transmit a message, LoRa was the best overall solution for indoor and outdoor scenarios, being the one that best adapts in terms of long distances and obstacles interference. ZigBee comes as a low power alternative for locations with fewer obstacles. It was also proved that using a higher transmission power does not always create the best communication link, with lower configurations being able to reach the same distances without affecting the link quality.

As for the point-to-point autonomous configuration system, a Machine Learning solution based on Random Forest was developed in order to predict the best protocol to use in terms of link quality and energy consumption. Associating this model with the system capabilities of deciding the best protocol based on the solution needs, either a higher reliability, lower power consumption or a middle approach, it was possible to achieve a system capable of adapting the protocol used to send a message based on the specification of the network and location, while saving 65% of the energy needed to exchange messages and only reducing by 13% the quality of the network. It was also possible to conclude that a

Edge Computing approach achieves better results, being quicker to reach a decision and saving more energy, when facing a Cloud Computing approach.

For the cloud communication autonomous configuration system, Edge Computing proved not to be capable of handling the decision part of the system, due to the limitations of the edge devices. So a Cloud Computing system was achieved using a Random Forest model, capable of predicting the link quality of wireless protocol to send messages to the servers, based on location, available protocols and near by gateways, and a scoring system, that chooses the best available protocols based on message size, communication cost, message limits and energy consumption. The system proved to be able to self-configure the node and adapt it, without human intervention, to communicate with the best possible protocol.

With the research presented in this chapter, several key points were gathered and developed for the final modular IoT solution system, as this will be heavily dependent on communication and message exchanges, allowing not only for an easier development of the next phases but also to guarantee consistency and reliability in those phases.

CHAPTER 5

Sustainable Smart Node

This chapter presents the research and developed solutions for the sustainable modular IoT smart node. It starts with an overview of the research, containing the related work found in the literature, followed by an explanation of some of the most important definitions, including a comparison of all the major IoT modules. With all the definitions introduced, this chapter continues with the system architecture, node features, modules and modes. It follows with detailed description on how modularity, self-configuration and sustainability were achieved through the use of Machine Learning and edge computing. The implementation of all the described methodology and features is presented. Finally, a remarks section closes this chapter with a brief discussion and conclusion of the topics and results given in this chapter.

5.1. Overview

Our approach on a true heterogeneous IoT solutions, is based on a fully adaptable Wireless Sensor Network (WSN), with a set of modular smart nodes that can adapt to the specification of the installation purpose. The system architecture follows the nature of a typical WSN, being the node capable of acting as three types of nodes (gateway, sensor and actuator). This is done using a plug&play modular fashion, where the necessary hardware modules are attached to the smart node in order to create a gateway, sensor or actuator node. This allows for a single node architecture to gather and transmit sensor data as well as perform tasks upon actuators in the environment.

To create the methodology for this smart node, research in order to find the best hardware components, not only in terms of the core controller, but also for communication modules, power supplies, sensors and other was necessary. In the first phase some off-the-shelf modules were tested, but the inability to make changes, some in hardware, others in software, leads us to creating our own nodes from scratch. The following methodology is the result of research and several iterations, based on tests, hardware availability and other specifications that will be described further.

5.1.1. Smart IoT Node

As described an IoT system must be capable of gathering, transmitting and analysing data and interact with the environment or user, in order to create a more autonomous and intelligent solution. For that it uses a set of devices, called smart nodes, capable of performing all these actions.

A smart IoT node is composed of several modules that when together create a device capable of gathering, transmitting and analysing information, through four main groups: Core, Sensors/Actuators, Power Supply and Communication [**115**]. Each of these modules create a layer that will help the node perform the needed tasks, as shown in Figure 5.1.

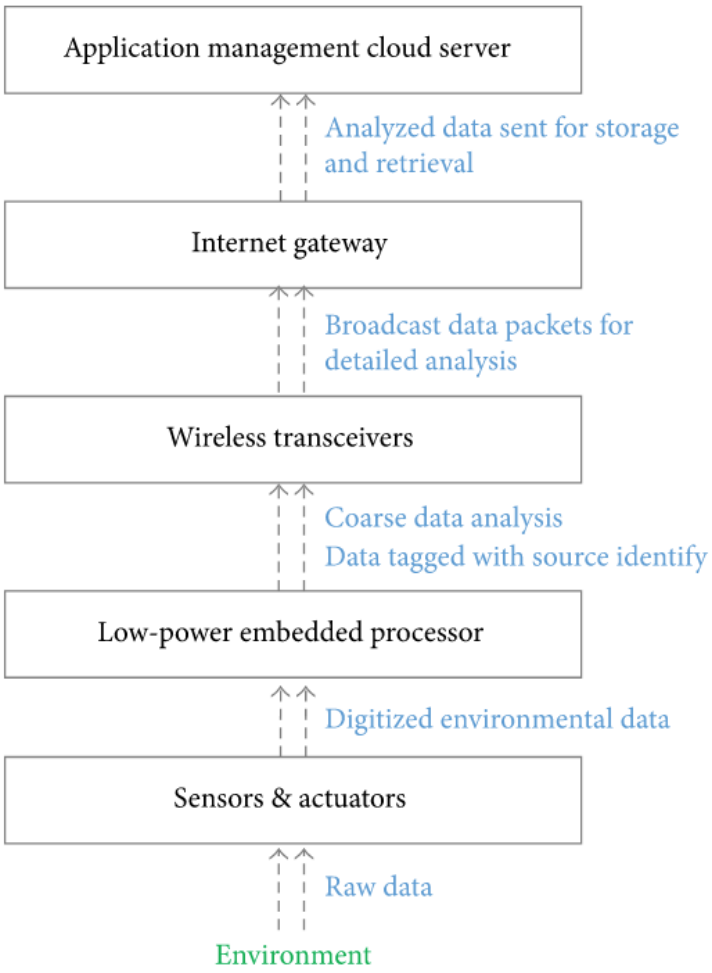


FIGURE 5.1. Smart IoT Node Layers

The Core includes all the computing power, composed by the microcontroller, memory and the needed connectors and hardware to merge with the other groups. The Sensors and Actuators are how the node interacts with the environment where it is implemented, by collecting sensor data or performing actions, with motors, lights or others. In terms of Power Supply, how the node is powered, this can be done in several ways, from typical

power transformers connected to the grid to batteries or self-sustained methods such as solar panels. Finally, the Communication is how the node will exchange data with other devices, and for that multiple wireless protocols can be used, as evaluated in the previous chapters.

Besides the physical part, an IoT smart node must have a behavior part that deals with all the needed functionalities, such as logic control, to connect all the modules, basic configuration, communication, sensing, acting, energy saving and update mode [**115**].

5.1.2. Core Microcontrollers

In the core of the smart node is usually a microcontroller, a small computer installed on a single integrated circuit chip that contains a CPU, memory and programmable input/output peripherals. Nowadays, with the proliferation of IoT systems and cheap electronics, the choice of microcontrollers is endless. Table 5.1 presents the most used microcontrollers in the market and their characteristics.

Table 5.1. Microcontrollers Characteristics

	ESP32 [**108**]	**ATMega32u4** [**116**]	**Cortex M0+** [**117**]	**nRF52840** [**118**]
Manufacture	Espressif	Atmel	ARM	Nordic
Architecture	32 bits	8 bits	32 bits	32 bits
RAM [kB]	520	2.5	8	256
Flash [MB]	4-16	0.032	0.032	1
Power Consumption [mA]	28	15	16	25
Power Consumption Sleep [μA]	10	12	3	1.3
Communications	BLE & WiFi	-	-	BLE & ZigBee
GPIO (ADC)	34 (18)	26 (12)	12 (4)	48 (8)

Since our smart node need to have as much communication protocols as possible, a considerable Flash memory size, for the edge computing models and a high number of available programmable pins, the ideal choice for the core microcontroller was the ESP32 ultra-low-power microcontroller [**108**], a dual-core chip with built-in WiFi and BLE, with embedded antennas, and the ability to connect up to 18 analog sensors.

5.1.3. Modular IoT Nodes

As described a smart IoT node must have a set of components to be able to function, including the Core, Sensors/Actuators, Power Supply and Communications. To achieve modularity each of these components must be exchangeable inside the node without compromising the functionality of the node. This is usually done using a plug&play system, where the modules are attached among them and can be swapped, removed or inserted without needing to make modifications to the main board.

Throughout the literature it is possible to found several approaches to modular and plug&play IoT nodes, from academic to industrial solutions. For example, both Arduino and Raspberry Pi projects follow this approach, where they sell a core board, containing the embedded processor, capable of being programmed and configured, and then sell a variety of shields that can be coupled with the core board to provide the remaining services, from actuator control to wireless communications. As such, they are able to provide a custom and modular solution that can fit the requirements of the system that can be changed by only swapping or adding a new shield, not requiring to change the entire hardware when a new feature is needed.

BITalino, a project developed by Hugo Silva et al. [**119, 120**], shows an academic approach to the modularity of smart IoT nodes, in this case for bio-signals acquisition and physical computing, that started in the university and is now commercially available. With their plug&play custom board they are able to create a device capable of retrieving data from sensors, transmit them over Bluetooth and communicate with a web application.

Other approaches were found that use off-the-shelf components to create modular and plug&play solutions for IoT smart nodes. In [**121**] the authors compare a set of off-the-shelf modules to create the best modular solution for an adaptable IoT solution that can fit in health monitoring, precision agriculture and indoor sensing. They found a wide choice of subsystems that are available in the market creating a set of solutions based on the system needs. Although this shows the possibility of creating a modular system it also presents some limitations, mainly with a wide variety of solutions being presented, falling in the lack of heterogeneous solutions problem, and for not presenting a true smart node, with cross-layer modularity not being presented, meaning that the nodes needed to be configured by hand and cannot be autonomous.

Several other concepts are shown to use custom boards instead of off-the-self components, such as [**122**] and [**123**], creating a true heterogeneous and modular solution, but none tackle the autonomous part and the interaction between modules.

Our goal and major contribution for a new smart IoT node is to create an architecture capable of being modular and to have autonomous configuration, based on module detection, error detection and edge computing.

5.2. System Architecture

Our approach on a modular IoT solution is based on a set of smart nodes capable of adapting to the needs of the WSN and their purpose, being adjusted with the help of Machine Learning. The way our approach achieves the desired adaptability is by using a smart node that can act as any of the three typical WSN nodes. This is done using a plug&play modular fashion, where the necessary hardware modules are attached to the smart node in order to create a gateway, sensor or actuator node, as shown in Figure 5.2, and edge computing analysis supported by Machine Learning algorithms, that analyse the modules attached and configure the node to act accordingly.

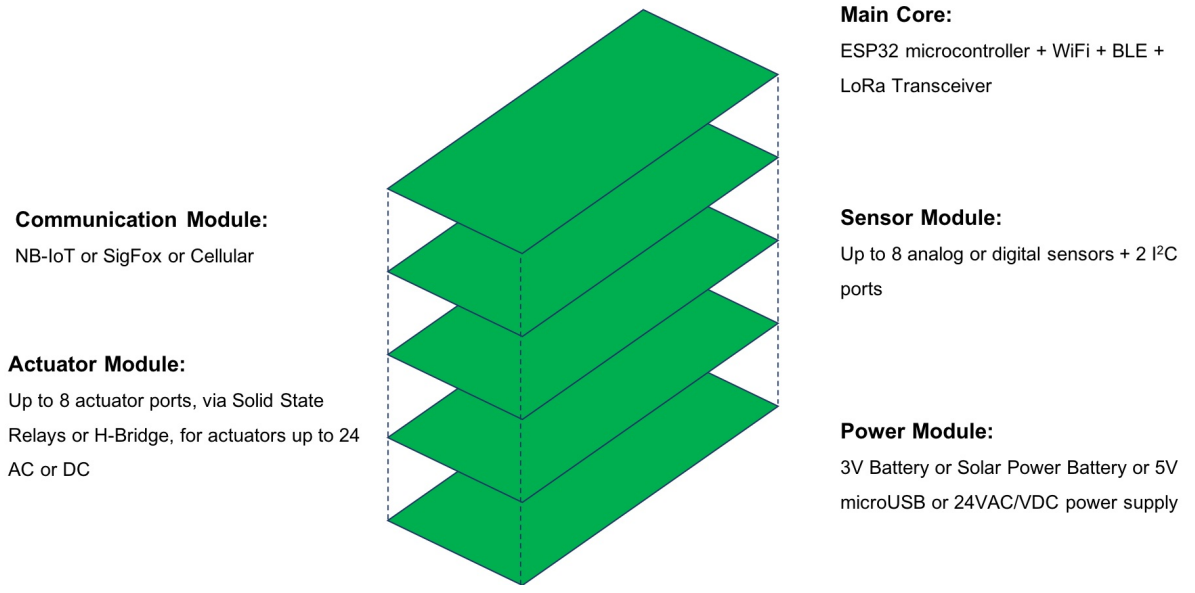


FIGURE 5.2. Node Overview

5.2.1. Node Features

As described before, the smart node differs from all the other IoT nodes due to its modularity and edge computing. Based on that, the node can act upon multiple specification, allowing it to perform a set of tasks to which other WSN application require multiple node architectures, such as:

- Create and manage the WSN: As the node can act as both gateway and sensor nodes, and with the ability to communicate between nodes, it can create a WSN with a star topology, with a gateway node in the middle managing the entire network, while other nodes act as sensor nodes and collect and send data to the gateway, as shown in Figure 5.3;

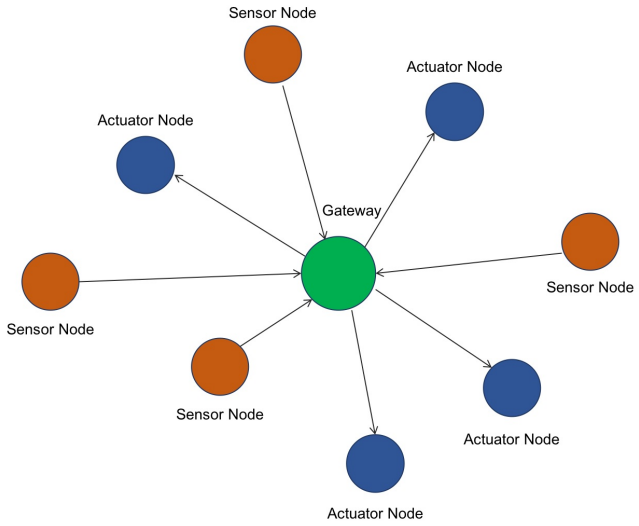


FIGURE 5.3. Star Topology

- Cloud computing: As the node can be equipped with several communication modules, it is able to communicate with cloud servers or platforms, such as Amazon AWS, Microsoft Azure, IFTTT, among many other, to store and analyse data;
- Edge computing: The node has the ability to analyse, in real time and locally, sensor data as they are collected by the sensor nodes, to allow for quicker responses to environment changes, based on Machine Learning models that can be trained and uploaded to the node;
- Information gathering: As the node can have multiple analog or digital sensors attached, it is capable of gathering information in real-time and transmit them;
- Actuator control: As for sensors, the node is also able to have multiple actuators attached, using relay systems to control high-power devices, such as motors or lights.

5.2.2. Node Modules

For these features to work, and as already specified, the node is composed of a set of modules that are attached to the main core module, in a Plug&Play fashion, to allow an

easy implementation. These modules include communication, sensors, actuators, power supply and other, and each of them will be explained in the following sections. Each of the modules includes a set of resistors, each with different values, that will be used to identify each module when they are connected to the main core.

5.2.2.1. ***Main Core.*** The main core of the smart node is composed by an ESP32 and a RFM95W LoRa transceiver [**110**], capable of creating an encrypted multi-point local network, ideal for communication between nodes inside the WSN.

5.2.2.2. ***Sensor Module.*** The sensor module is composed of an 74HC4051, an 8-channel analog multiplexer, capable of reading up to eight analog sensors while using one analog port from the main core, freeing more ports for other features. This module has four analog sensor ports, each with a VCC, Signal and GND line, and two I^2C ports, each with the SCL, SDA, VCC and GND lines, that are directly connected to the ESP32 in the main core. If more sensor ports are needed, the sensor module has an expansion module in order to add four more sensors and two I^2C ports.

5.2.2.3. ***Actuator Module.*** The actuator module is composed of an 74HC595, an 8-bit serial-in, parallel-out shift register, that uses three digital ports from the main core to control up to 256 digital outputs, once again, freeing more ports for other features. Since the ESP32 might not have enough power to supply the coupled actuators, this module is equipped with four Panasonic AQY212EHAT Solid State Relays (SSR) in order to operate, through a simple electronic circuit and HIGH and LOW signals from ESP32, actuators up to 60 VAC/VDC. Each of the SSR also has a Resistor/Capacitor snubber, to protect the module against short-circuits. This module includes a power entry, to supply the needed current to power the actuators. If more than four actuators are needed, this module has an expansion module to add four more actuator ports or more eight, if needed.

5.2.2.4. ***Communication Module.*** In the main core, the node already has access to WiFi, ESPNow and BLE, from the ESP32, and LoRa and LoRaWAN from the RFM95W, but to ensure that the node can actually adapts to any specification, and work with the developed autonomous communication configuration system, several communication modules were developed, including 2G/3G/4G, using the SIM7600E, NB-IoT, using the SIM7000E, and SigFox, using the ATA8520E, for the cloud communications, and ZigBee, using the XBee Pro3, for point-to-point communication. For each, all the necessary

hardware to put the communication module to work and the corresponding antenna is included.

5.2.2.5. *Power Module.* The power module is divided into four categories, each of them design to fit with a set of other modules:

- Battery Power: Includes a CR123A 3V battery holder, capable of powering the main core and sensor module and is capable of reading the battery capacity;
- USB Power: Includes a micro-USB entry and a LM1117-3.3V LDO regulator to convert the 5V power supply to the needed 3.3V;
- Solar Power: Includes the TP4056 battery charge module and the ports to connect the solar panel and a lithium 3.7V battery that will store the solar energy. It also includes a LM1117-3.3V to convert the 3.7V power supply to the needed 3.3V;
- AC or DC Power: Includes a LM2596, a step-down switching regulator, to convert the input power, min. 9V to max. 24V, to 5V and then to 3.3V using the LM1117-3.3V. It also includes a KBP206G bridge rectifier to convert the AC power input into DC power, if needed.

5.2.3. Node Tasks

Although the same core is used on every type of node, to accomplish the specific tasks required by each node inside the WSN, and allow it to work as Figure 5.4 shows, the smart node adapts according to the modes presented in the next sections.

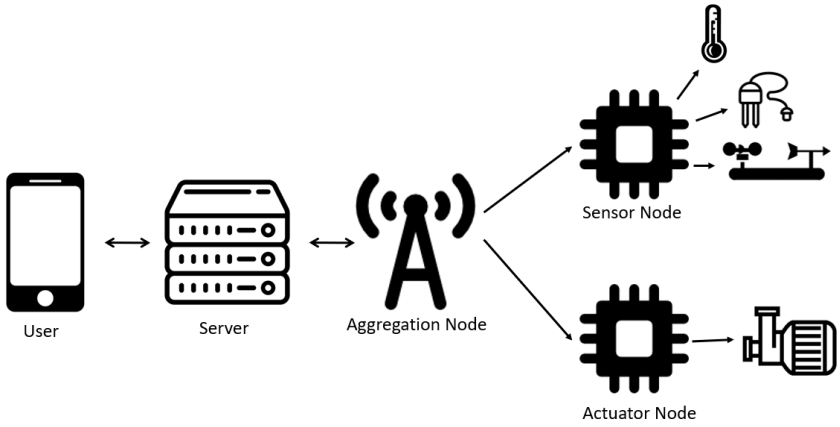


FIGURE 5.4. WSN Node Tasks

5.2.3.1. *Gateway Mode.* Only the main core is used, attached with the USB power supply module. In this scenario the developed autonomous cloud communication system,

from Section 4.4, is used to connect the WSN to the cloud. The roles of this node is to create and control the entire WSN point-to-point network, receiving messages from the other nodes and sending them to the cloud, as well as receive messages from the cloud and send them to the corresponding node, using the autonomous point-to-point communication system, in the BLM configuration, from Section 4.3.

The node-to-node communication is how information is exchanged inside the WSN, from sensor information to actuator actions. Since nodes can be far from each other, and as some of the nodes are powered by batteries, as concluded in Section 4.2, LoRa is the ideal solution. The proposed system can support a network of up to 250 nodes, with individual node addressing, with bigger networks not compromising the reliability of the system nor giving extra work in the implementation, since the nodes are configured automatically inside the network when booting for the first time.

The node-to-server communication is how information containing sensor data leaves the network and actions are received. In order to create a constant low-power data connection, the MQTT protocol was adopted, since it is an optimal connection protocol for IoT and M2M, being suitable for small, cheap, low power and low memory devices with low bandwidth networks. Being built on top of the TCP protocol, and using a publish/subscribe pattern, is capable of providing flexibility and simplicity connectivity [**124**].

5.2.3.2. *Sensor Mode.* The main core is used, attached with the sensor module and the battery or solar power module. This mode is only responsible for gathering sensor data and sending them to the gateway via LoRa, including also checking the battery capacity and sending a message when less than 30% capacity is reached. Since it is a battery powered node, when the node is not collecting data it enters in a deep-sleep mode, allowing the battery to last longer. In this mode, the autonomous point-to-point communication system, in the RLM or EFM configuration, is used to transmit the collected data. It can also implement some edge computing data analysis.

5.2.3.3. *Controller Mode.* Attached to the main core is the controller module, and the AC or DC power supply module, powered by an external power supply. This node is responsible to perform tasks in the installation sites, such as turn ON/OFF actuators, when a new message arrives from the gateway. For that it uses the autonomous point-to-point communication system, in the BLM configuration. In some implementations, and since this node is always powered by a fixed outlet, the mode can also include the sensor

module, in order to gather information, analyse them directly on the node using edge computing, and act upon the actuators in real-time.

5.3. Edge Computing & Self-Configuration

As said, the developed smart node is autonomous and can be self-configured without human intervention, as it is supported by a set of edge computing algorithms. These algorithms are responsible for detecting which modules are attached to the smart node, if they are compatible, mainly in terms of power supply, and finally configuring the node workflow according to the tasks each module requires.

Figure 5.5 represents the step by step analysis done by the edge computing when the node is turned ON. It starts by retrieving the values from the modules attached to smart node, using them to predict, with a Random Forest Classifier, the type of module; then, knowing all the modules attached and their requirements, it will detect if there is any incompatibility between the modules; and finally, if no errors are detected, it will identify the node as a gateway, sensor or controller node and configure the node for the corresponding tasks.

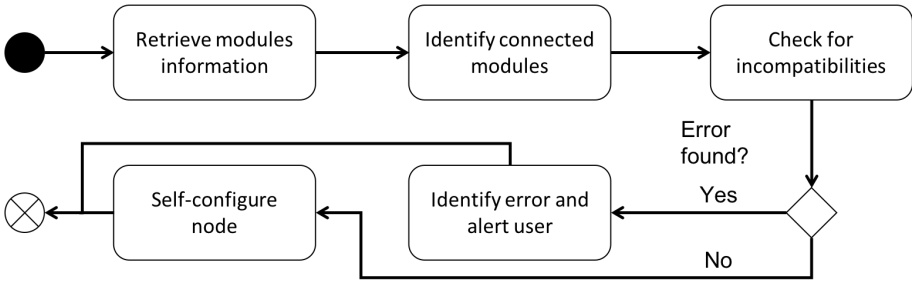


FIGURE 5.5. Smart Node Edge Computing Methodology

Each of these steps will be described in detail, including how each of the edge computing models were developed, in the following sections.

5.3.1. Module Detection

The first thing the node does is retrieve information about the connected modules, using the resistance values of each module, that are collected by the main core. Those values are collected using an 74HC4051, being each of the input ports dedicated to a specific module, as described in the Module ID column of Table 5.2. Then the goal is to predict which module is attached to that specific port, as described on the Module Option and Description column in the same table.

TABLE 5.2. Module Codes for Module Detection

Module	Module ID	Module Option	Module Description
Sensor	0	0	Not Available
		1	Sensor Module (4 sensors + 2 I^2C)
Actuator	1	0	Not Available
		1	AC Actuator Module (4 actuators)
		2	DC Actuator Module (4 actuators)
Communication	2	0	Not Available
		1	NB-IoT Module
Power	3	0	Not Available
		1	Battery Power
		2	AC Power
		3	DC Power
		4	USB Power

To create an edge computing model capable of predicting the module type based on the resistance value, the training methodology followed across all our research, and described in Section 3.2.1, was followed, being Random Forest the selected algorithm.

To create the training dataset, each combination of the modules was assembled and the resistance values were retrieved 500 times, in order to create a dataset capable of identifying the modules as best as possible. The final dataset contains 6000 entries with the following parameters:

- Module ID – Specification of the module type
- Module Option – Variation of the module type
- Value – Resistance value

The training methodology presented in Section 3.2.1 was followed in order to obtain the best Random Forest classifier model possible to predict the attached module, based on the module resistance.

Following the methodology, the hyper parametrization tuning, showed that the best configuration for the Random Forest classifier model was with the following parameters:

- n_estimators – 10
- max_features – 'auto'
- max_depth – 10
- min_samples_split – 5
- min_samples_leaf – 4

- bootstrap – True

This model achieved an accuracy of 64.66%. Figure 5.6 shows the predicted values versus the real values, obtained by the model.

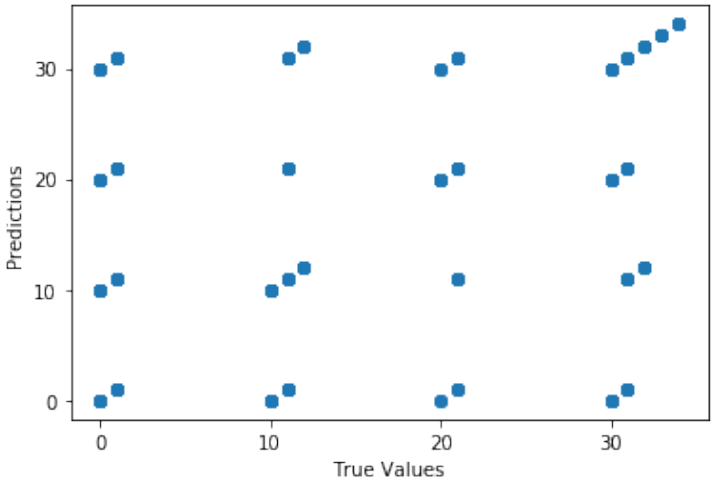


FIGURE 5.6. Module Detection Predicted vs Real Values

To further validate the model accuracy, and following the methodology presented, the model was validated with a Stratified K-Fold Cross Validation, using 5 folds and 20% of the data as validation points. Figure 5.7 shows the learning curve for the validation test.

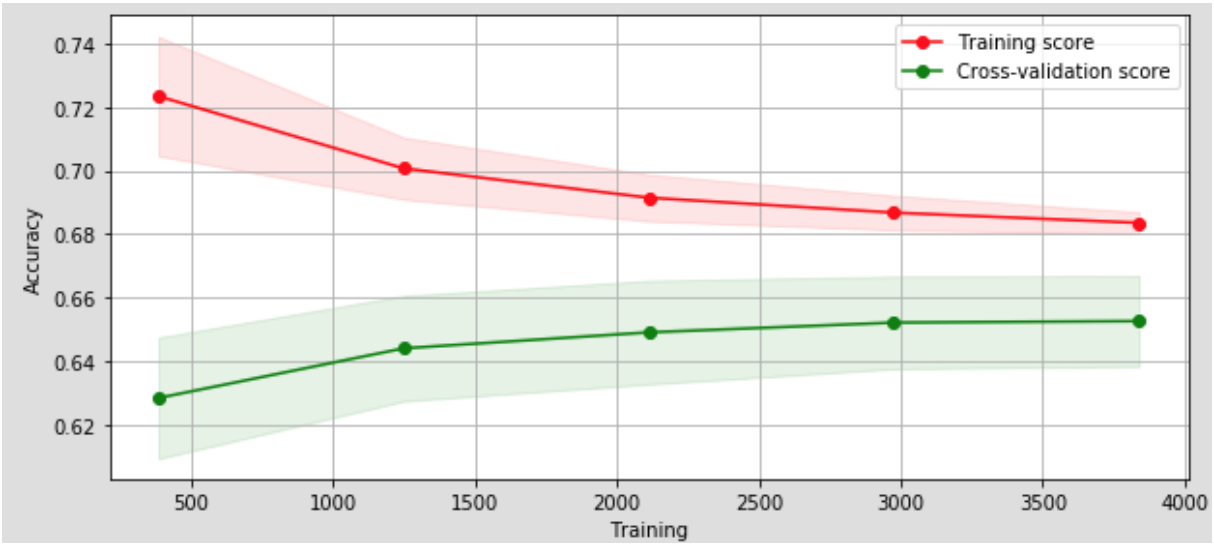


FIGURE 5.7. Module Detection Classification Learning Curve

It is possible to observe, at the end of the learning curve, that both training and validation converge to a similar value, showing that the model is well fitted, with a validation accuracy of 65.2%.

As such, is it possible to conclude that the trained model is not suitable for predicting the attached module based on its resistance. In a further evaluation of the trained model,

the results matrix, Figure 5.8, was analysed to check if it was possible to understand what was happening.

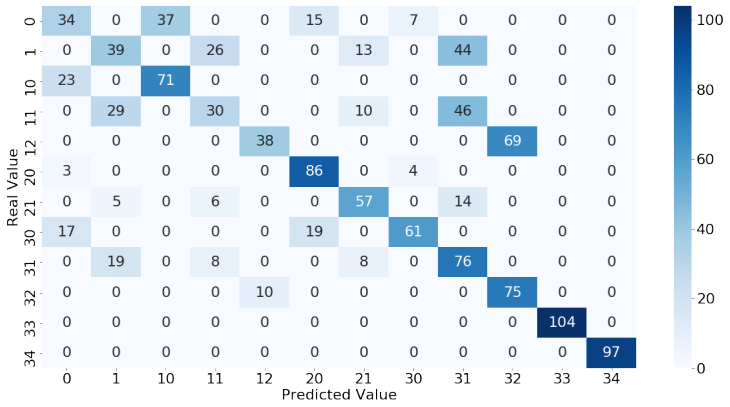


FIGURE 5.8. Module Detection Matrix Analysis

Analysing these results and considering the format of the intended output, that is composed by two numbers, the first identifying the Module ID and the second the Module Option, in the matrix is possible to assess that the second number is always correctly identified, being the problem with the identification of the Module ID. But, due to the following methodology for the module detection, the Module ID is already identified by the main core when choosing the input port of the 74HC4051, so the first number from the classification model result can be discarded and replaced by the main core identifier. This logic is used as follows:

```
for(i = 0; i < 5; i++){
    // Read the input value from the port i of the 74HC4051
    v = readModuleValue(i);
    // Use the edge computing model to predict the module
    // classification
    p = model.predict(v);
    // Remove the first bit from the result and append the
    // input value
    r = i + strip(p, lastbit);
}
```

With this revised methodology implemented alongside the classification model, the implementation was tested by attaching several modules to the main core and checking if the output corresponds to the module attached. For each combination, 100 tests were

performed, in order to guarantee the reliability and accuracy of the detection system. Figure 5.9 shows that the revised methodology is able to achieve 100% accuracy in detecting which modules are attached to the main core.

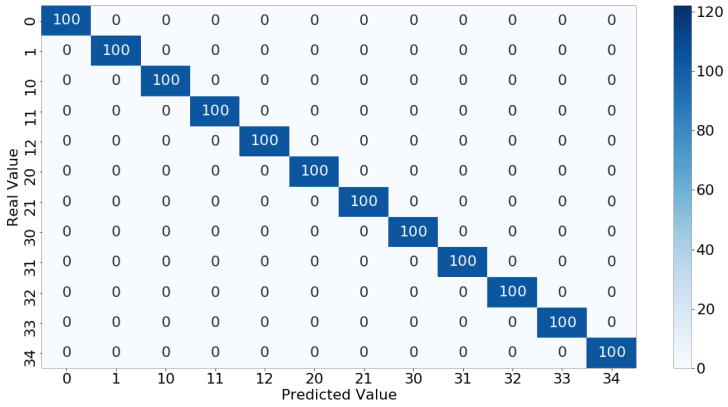


FIGURE 5.9. Revised Module Detection Matrix Analysis

With a validated and accurate system for detecting which modules are connected to the main core, the Random Forest model was ported to run on the edge device. The ported model created a file with 334 kB, so no modifications to the model were needed to be able to port it to the microcontroller.

5.3.2. Configuration Error Detection

After knowing which modules are attached to the main core, the next step in the self-configuration system is to detect if there are any incompatibilities between the modules, mainly in terms of the power source needed for each mode. Table 5.3 shows the possible outputs from the modules combinations.

TABLE 5.3. Configuration Error Output

Error	Description	Combinations
1	No power source	No power module attached
2	Incompatible power source	Actuator with or without Communications with Solar, Battery or USB
		Any cellular communication with Solar or Battery
0	No error	Other combinations

To create an edge computing model capable of detecting incompatibilities in the attached modules, the training methodology followed across all our research, and described in Section 3.2.1, was used, being Random Forest the selected algorithm.

The training dataset is composed of each of the possible combinations of modules and the corresponding output, as presented in Table 5.3, in order to create a dataset capable of identifying errors as best as possible. The final dataset contains 168 entries with the following parameters:

- Sensor Module – Module Option for the Sensor Module (see Table 5.2)
- Actuator Module – Module Option for the Actuator Module (see Table 5.2)
- Communication Module – Module Option for the Communication Module (see Table 5.2)
- Energy Module – Module Option for the Energy Module (see Table 5.2)
- Error – Error output (see Table 5.3)

The training methodology presented in Section 3.2.1 was followed in order to obtain the best Random Forest classifier model to predict the attached module, based on the module resistance.

Following the methodology, the hyper parametrization tuning, showed that the best configuration for the Random Forest classifier model was with the following parameters:

- n_estimators – 15
- max_features – 'sqrt'
- max_depth – 10
- min_samples_split – 5
- min_samples_leaf – 2
- bootstrap – False

This model achieved an accuracy of 100%. Figure 5.10 shows the predicted values versus the real values, obtained by the model.

To further validate the model accuracy, and following the methodology presented, the model was validated with a Stratified K-Fold Cross Validation, using 5 folds and 20% of the data as validation points. Figure 5.11 shows the learning curve for the validation test.

It is possible to see, at the end of the learning curve, that both training and validation converge to a similar value, showing that the model is well fitted, with a validation accuracy of 89.5%, .

As such, is it possible to conclude that the trained model is suitable for detecting incompatibilities in the attached modules. The 100% accuracy in this model can be justified by the dataset containing all the possible scenarios and not much variation can

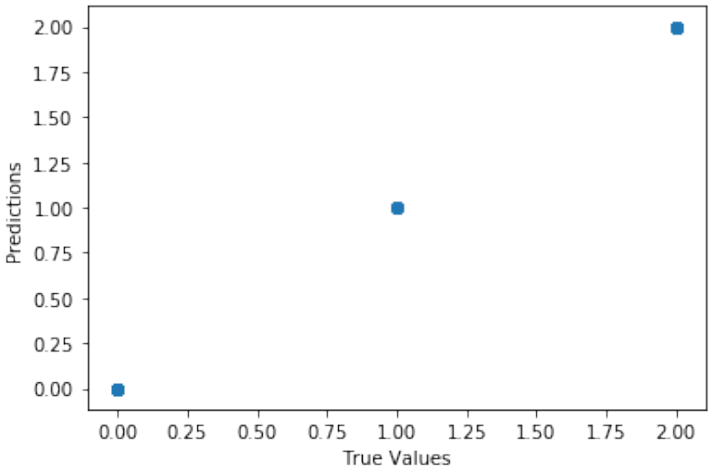


FIGURE 5.10. Error Detection Predicted vs Real Values

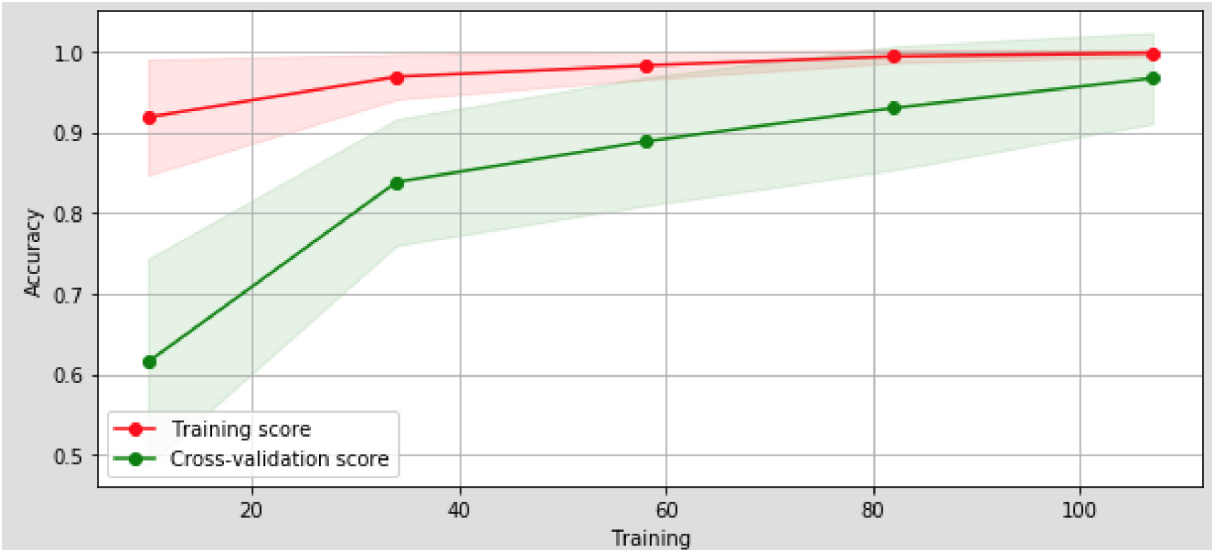


FIGURE 5.11. Error Detection Classification Learning Curve

be found in a real case scenario, so it is normal for the model to have a perfect fit and accuracy.

With a validated and accurate system for detecting incompatibilities in the attached modules, the Random Forest model was ported to run on the edge device. The ported model created a file with 127 kB, so no modifications to the model were needed to be able to port it to the microcontroller.

5.3.3. Configuration Mode Detection

If no errors are found in the previous step, the last step of the self-configuration is to configure the node with the corresponding tasks based on the attached modules. For this step, a simple switch case scenario was implemented, since it is only a matter of checking if a module is attached and to implement the needed features.

Based on the attached modules, the following modes are configured in the node:

- Only the Main Core – The Gateway Mode is configured. The node will create the LoRa network, connect with the server, using the autonomous cloud communication system, and start listening for other nodes;
- Sensor Module – The Sensor Mode is configured. The node will gather information from the attached sensors and send them to the gateway. Deep sleep can be configured in between the collection periods;
- Actuator Module – The Controller Mode is configured. The node will wait for messages from the gateway and control the actuators based on those messages;
- Battery or Solar Module – When one of these modules is attached, a battery capacity check is performed, in order to send a message when the capacity falls under 30% and warn the user that it might need to be replaced;

In the sensor or controller mode, it will broadcast a message saying a new node was created and wants to be part of the network. After being accepted into the network, it starts the corresponding tasks. In order to be fully configured, the user then needs to say what type of sensors or actuators are attached, so the software can understand the data that is being collected and how to control those actuators. This is the only human interaction needed to create the system.

5.3.4. Data Analysis

For the Data Analysis using Edge Computing models, as shown in the previous chapters, there are already a set of models that need to be part of node, such as the autonomous communication systems and the module detection and self-configuration system. Table 5.4 shows the space already allocated for these decision models.

Table 5.4. Edge Computing Allocated Space

Function	**Model**	**Size [kB]**
Node Configuration	Module Detection	334
	Error Detection	127
Communications	Point-to-Point Configuration System	940
	Total	1401

Since the ESP32, the microcontroller used in our Main Core, has 2 MB of available Flash to store edge models, it is still possible to implement an edge computation data

analysis system in any of the node modes, regarding that the model file needs to have less than 600 kB.

5.4. Implementation

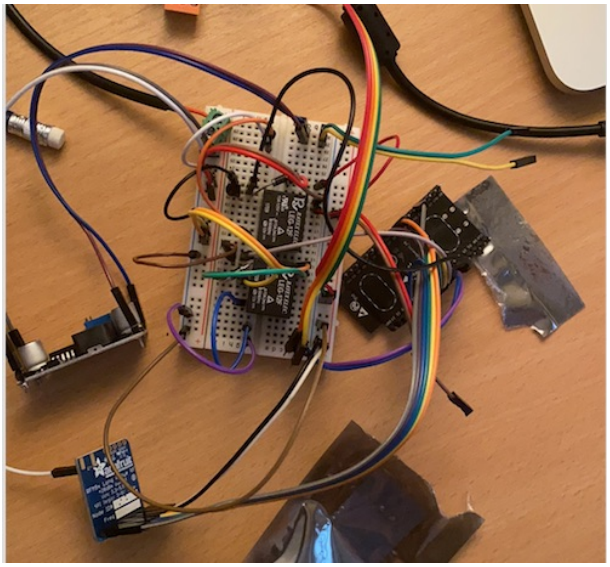
As said, in order to create our smart node some research was done to understand which were the best components to use and how to connect them into the final solution. For that, the development and implementation of the smart node started with some off-the-shelf modules assembled in a laboratory environment and only after that they were ported into custom Printed Circuit Board (PCB) boards.

In a first stage each of the nodes, aggregation, sensor and actuator, were assembled using the ESP32-DevkitC-v4, using jumper wires and a breadboard to simulate the modularity. The goal was to combine all the needed hardware and understand the needed pins for each module and all their variations. Figure 5.12 shows the assembled gateway, with the NB-IoT communication module, the actuator, with the relay system, and the sensor node, with the 74HC4051 and sensor connectors.

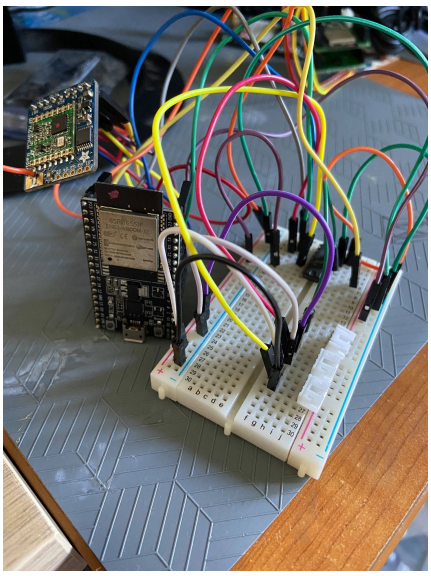
After understanding how the modularity will work and the needed pins and components, the ESP32-DevkitC-v4 was replaced with a bare ESP32-WROOM-32E module, using an adapter board, Figure 5.13, and only the necessary components. This step was crucial, since the ESP32-DevkitC-v4 had several components that were not needed in our implementation, such as LED's, voltage regulators, USB connectors and several resistors and capacitors that not only increase the cost of the device but also consume unnecessary energy when the node is connected. With this step it was possible to understand how the core module can be ported into a custom node

The final step was to develop a custom PCB board for each of the modules, as described in Section 5.2.2. For that, the KiCaD software was used and the PCBs were manufactured in JCLPCB, a Chinese PCB manufacturer. Figure 5.14 shows the final modules for the Core, Sensor, Actuator, DC Power, AC Power, USB Power and Battery Power. The schematics, PCB layouts and BOM files can be found in Appendix C

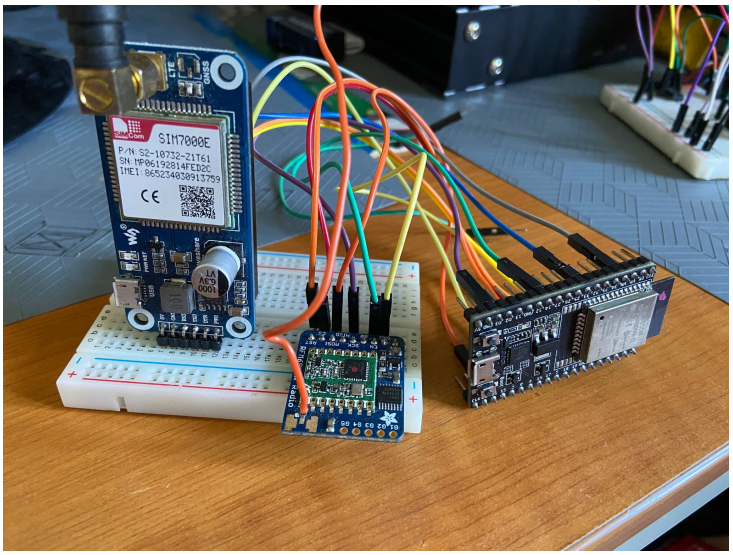
The modules can be stacked on top of each other in order to connect them and create the node modes described in Section 5.2.3.



(a) Actuator Node



(b) Sensor Node



(c) Aggregation Node

FIGURE 5.12. Breadboard Implementation

FIGURE 5.13. ESP32 Adapter Board

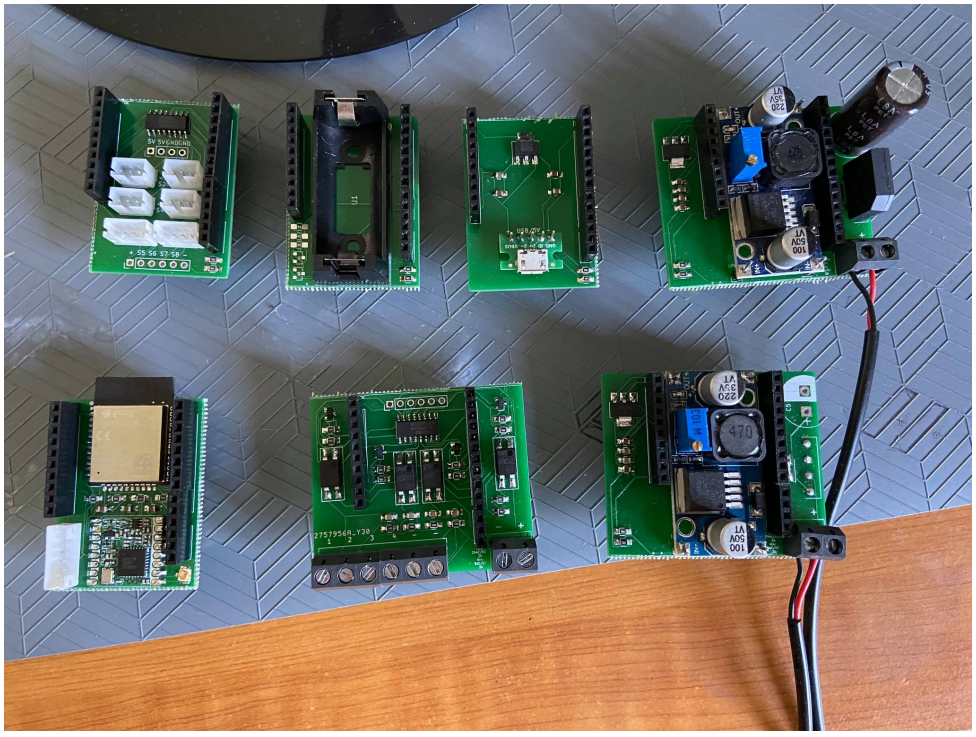


FIGURE 5.14. Modular IoT Smart Node

5.5. Dashboard & Control

Although a heterogeneous WSN is presented, capable of adapting to the specifications of the installation without human intervention, there is still a need to tell the network what those specifications are. Also, any IoT system requires a visualization platform, capable of showing the data collected by the network as well as give the user a way to send commands to the network when needed.

Our research shows that when the typical user has an application for each of the systems they use, mainly when they are from different manufactures. Even when the same company offers two IoT systems to the user, they use different apps or dashboards to control them.

Since our goal was to provide a heterogeneous solution, not only in terms of hardware but also in software, with our interactive dashboard the user can control their networks regardless of the specifications or environment. The dashboard will act as gateway between the WSN and the user and has the following features:

- Cross platform, i.e., available on web, Android and iOS;
- Allow remote control over the network, regardless of the location of the user or the network;
- Visualization of all the collected data;
- Basic configuration of the network:
  - Identifying the attached sensors – This is how the network will be able to self-configure the nodes;
  - Identifying the attached actuators;
  - Identifying the purpose of the network – This is how the software will be able to analyze the sensor data;
  - Remove or disable nodes from the network;
- Manually control the actuators;
- Receive alarms;

The dashboard will show all the sensors connected to each network, showing all the corresponding data and configurations. Whenever a new node is connected, it will automatically appear in the dashboard and be ready for the user to conclude their configuration. Figure 5.15 shows the developed dashboard on all platforms.

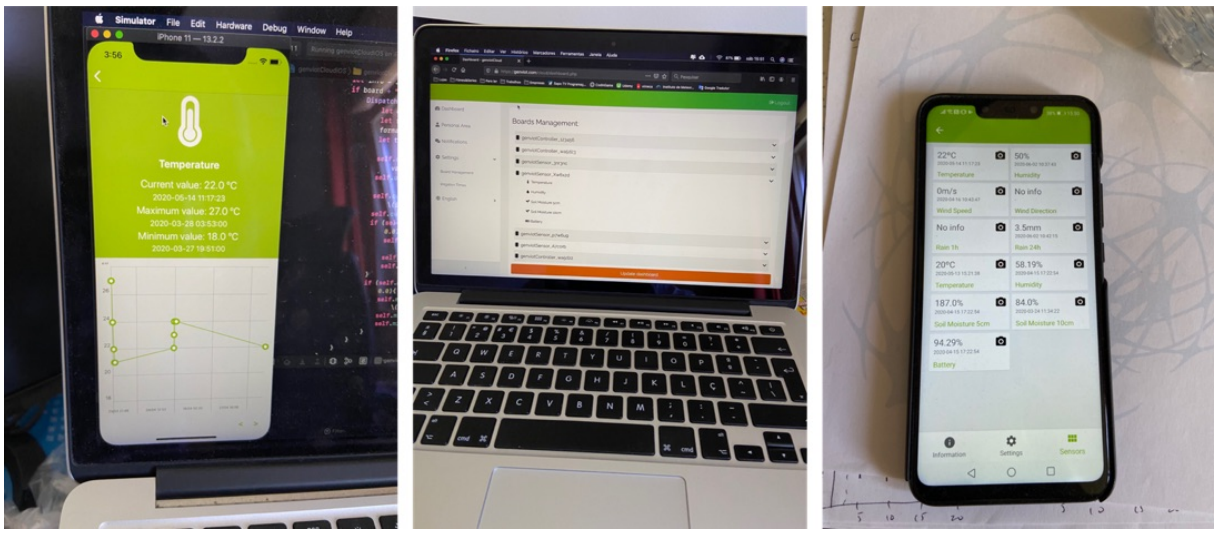


FIGURE 5.15. System Dashboard

5.6. Remarks

This chapter presented the research and development of the sustainable modular IoT smart node supported by Machine Learning. From the introduction of the needed concepts to develop a modular IoT smart node to the developed intelligence that controls the node it shows how the smart node was built, its features and implementation.

With a goal of creating a modular node, several modules were designed in order to create a node that can act as a gateway, actuator or sensor node, while being powered by a set of different ways, such as batteries, solar or DC power. Based on the used modules and the combinations between them, tasks were created for the model to act upon.

As the module will be self-configurable based on the attached modules, several edge computing models were created based on the Machine Learning methods. For detecting the attached modules a Random Forest Classifier capable of assessing the modules based on its resistance, achieving a low accuracy of 65%. After a careful analysis of the model results, using a simple logic based on the previous results allows the model to improve its results and achieve a 100% accuracy when detecting the attached modules based on its resistance. After knowing what modules were attached, it was necessary to know if there was some configuration error, i.e., if any of the attached modules were not compatible. For that a Random Forest Classifier was developed to identify incompatibilities among the attached modules based on each module's ID. This model achieved an 89.5% accuracy. Finally, in order to self-configure the node, a script was created to start the needed tasks based on the attached modules. All these intelligence and computing models were ported to the node in an edge computing fashion.

As the smart node is composed of a set of different modules, each of them was designed into a custom PCB and manufactured. Each of the modules was created using only the bare and needed components, in order to create the most energy efficient module.

The proposed hardware architecture for the WSN meets the requirements of the current IoT systems and proves to be an optimization from the current solutions on the market, since it is able to adapt to multiple situations or scenarios. Besides that, the ability to perform in a low-power scheme and also adapt the consumptions to the user needs, allows the system to be very reliable in terms of battery usage.

CHAPTER 6

Implementation & Case Studies

This chapter presents the implementation of the developed sustainable modular IoT solution for Smart Cities application supported by Machine Learning. It starts with an overview of the case study scenarios and guidelines for the performed implementation tests. It follows with the description of the implementation scenarios where a traditional IoT system was already implemented, including the problem statement, the system description and the obtained results using the current implementation. For each of the implementation scenarios, the way our new solution is implemented is described as well as the obtained results. Each scenario ends with a comparison between the tradition and our new approach in terms of efficiency, energy used and decision time, in order to understand if our solution is capable of having the same performance as a typical solution while creating a more sustainable process. Finally, a remarks section closes this chapter with a brief discussion and conclusion of the topics and results given in this chapter.

6.1. Overview

Cities depend on water to achieve a successful environment, not only for human consumption but also for the production of food and materials, green spaces and agricultural field irrigation and many others. But most of these activities still use humans to administer the correct amount of water and manage the entire systems, leading to complex and inaccurate results.

For example, in irrigation, one of the major activities in terms of water consumption, with more than 70% of the world fresh water being used for landscape and agricultural irrigation, 30% of that water is wasted or misused due to many situations such as lack of control, leaks or misuse, according to [**125**].

As such, we will implement our new sustainable modular IoT solution facing two developed systems during our research, in the field of water management in Smart Cities applications: detecting leaks in water distribution pipelines; and improving the water management in irrigation systems for both landscape and urban farming.

In each of these applications, the previous system will be described, with the corresponding results, as well as the new implementations and obtained results, being then compared among them.

6.2. Leak Detection for Water Distribution Pipelines

To achieve the main objective of detecting leaks and their location in pipes in real time using Machine Learning, there was a need to design a control and monitoring system to be applied in water distribution pipelines in the public and private domains. This system intends to use multiple sensors with real-time data collection, mainly water flow parameters, to improve the efficiency and the early detection of leaks, with the support of Machine Learning techniques.

6.2.1. Traditional IoT Approach

The system consists of various water flow measuring sensors, spread throughout the water distribution pipeline to collect information as water flows by them [**126**]. Each of the sensor nodes, Figure 6.1, consists of an ESP32-DevKitC microcontroller and a YF-B2 water flow sensor [**127**], a water rotor combined with a hall-effect sensor, that detect speed changes with the different rate of water flows through it. To transmit the gathered sensor information to the aggregation node, it uses a LoRa connection through the RFM95W module. This node is powered by an 3V CR123A battery that, combined with a deep sleep functionally of the ESP32, can last up to 545 days.

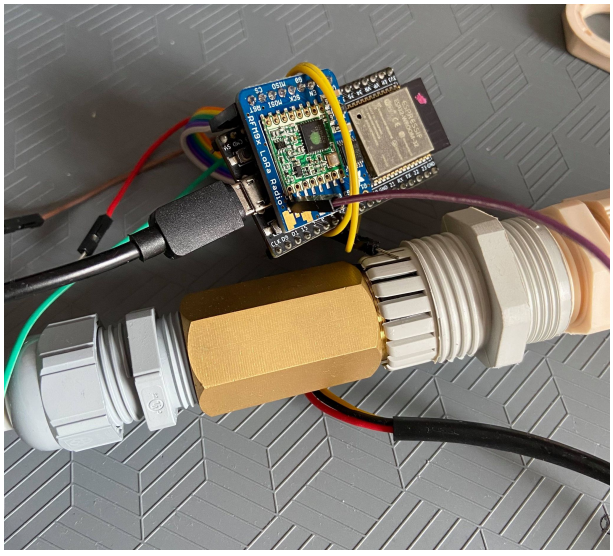


FIGURE 6.1. Leak Detection Traditional Sensor Node [**126**]

The aggregation node, Figure 6.2, is responsible for the communication with the server, transmitting the information collected from the sensors. It is composed of an ESP32-DevkitC and a RFM95W LoRa module, to receive the messages from the sensor nodes. For the server communication, MQTT is used via an NB-IoT connection, using the SIM7000E module.

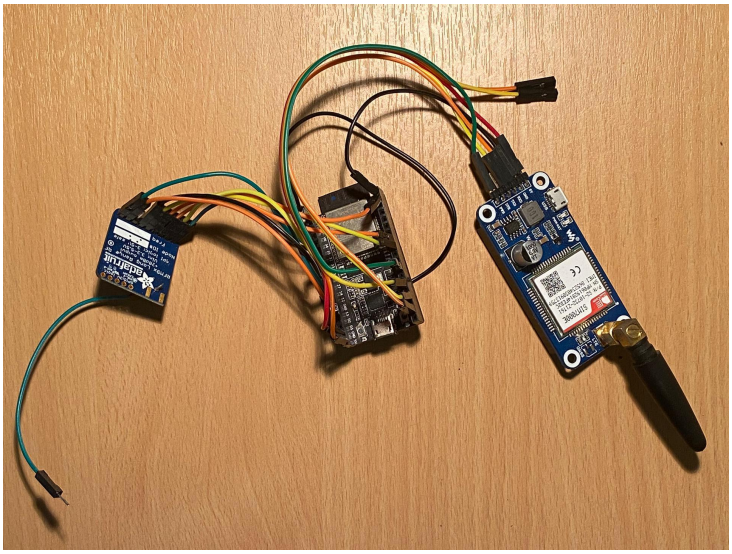


FIGURE 6.2. Leak Detection Traditional Aggregation Node [**126**]

In the server, the information is stored and then analysed using a Random Forest Classifier, that through the previous study achieved an 85% accuracy predicting water leaks and to inform the user of leak location, based on the sensor data. Depending on the analysis done by the algorithm, the information is shown to the user as being all normal within the system or will alert the user for potential locations of water leaks. This logic can be better comprehended by the flowchart of Figure 6.3.

The system was implemented in a simulation scenario that mimic a set of pipelines that supply water in irrigation systems or households, with the goal of warning the user when situations such as leaks start to appear and their location, not only to notify the user but also to help prevent this type of situation from evolving to bigger ruptures or other problems, such as water and monetary waste. Figure 6.4 shows the experimental laboratory implementation of the water distribution system.

The system was left running for several hours, collecting data from individual sensors as water flowed through them and sending those values to the server to be analyzed and to predict whether leaks are occurring and where they are. The test started with a new set of pipes in perfect conditions and, over time, holes were made in them to simulate leaks. As such it is possible to check if the system can detect them and warn the user

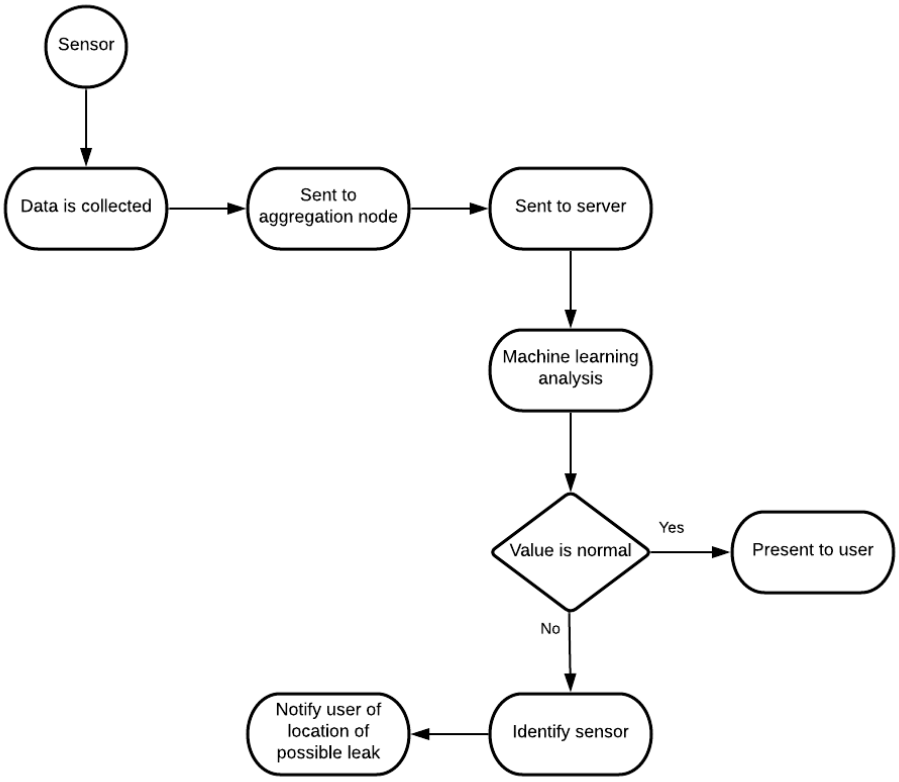


FIGURE 6.3. System Logic [**126**]

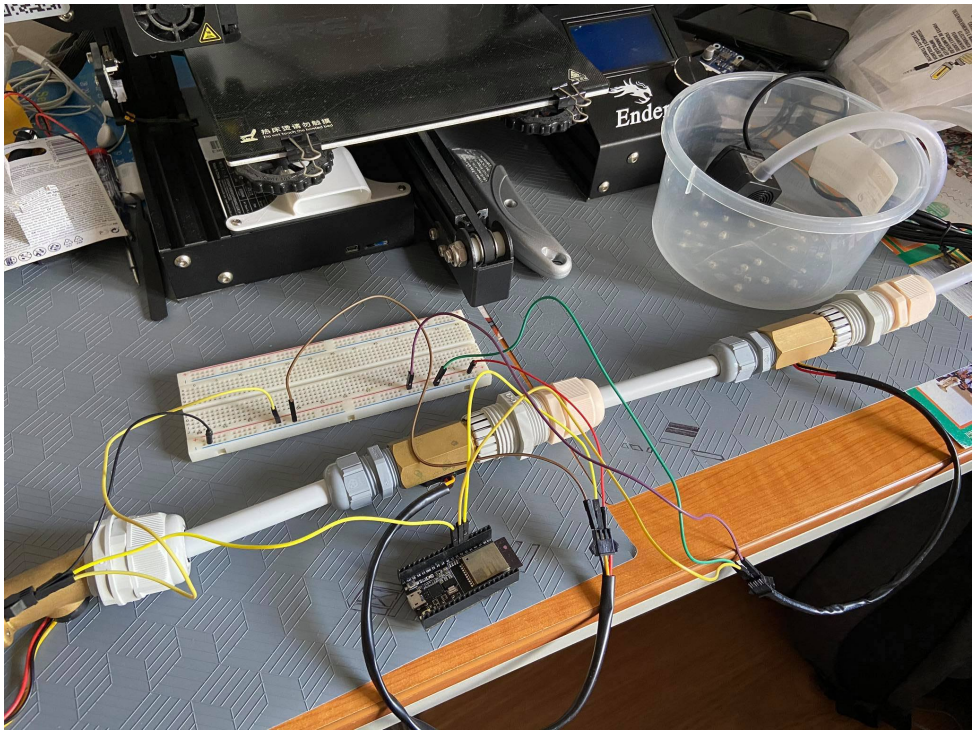


FIGURE 6.4. System Implementation [**126**]

about their location in real time. To check the accuracy of the system, as each hole was

made, the timestamp and location were recorded to later compare with the results from the system.

Figure 6.5 shows the obtained results, allowing us to observe when true positives, true negatives, false positives, and false negatives were obtained for each of the possible outputs. With this, it is possible to understand the situations where the outputs were wrongly predicted and to calculate the accuracy of the system for that particular test.

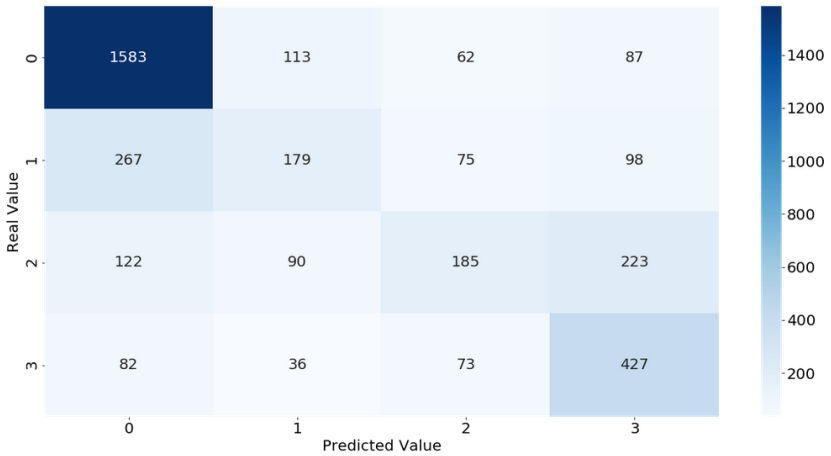


FIGURE 6.5. Traditional Implementation Results [**126**]

It is possible to see that the predicted outputs of the system are able to detect not only if a leak exists but also the section where it occurs based on the data collected from the sensors. Looking at the diagonal in the matrix that presents the true positives, where the values were correctly predicted, it shows that the system is able to detect more correct situations than wrong ones, with 2374, accounting for a 75% accuracy. There are some mistakes between the 0 and 1 output values, indicating that, when detecting minor leaks, the system still needs to be improved. Also, some situations where the output was 2 were identified as 3. The more concerning situations are major leaks that were identified as non-problems in 82 cases and situations where no leak was present but was identified 87 times as majors' leaks. This showcases that the accuracy of the system is still not perfect but is in line with the results from the training, where the RF model got only 85% accuracy.

6.2.2. Sustainable Modular Approach

Following the system architecture from the traditional approach, the nodes were replaced by our sustainable modular approach and the Machine Learning model was implemented in an Edge Computing model, directly in the nodes, instead of the Cloud Computing model in the server.

For the Aggregation Node, the Gateway Mode, described in Section 5.2.3.1, was used. Composed by the Main Core and the USB Module, it is capable of gathering the LoRa messages and sending them to the server using MQTT. For that, the NB-IoT Module was also attached. Figure 6.6 shows the modular aggregation node.

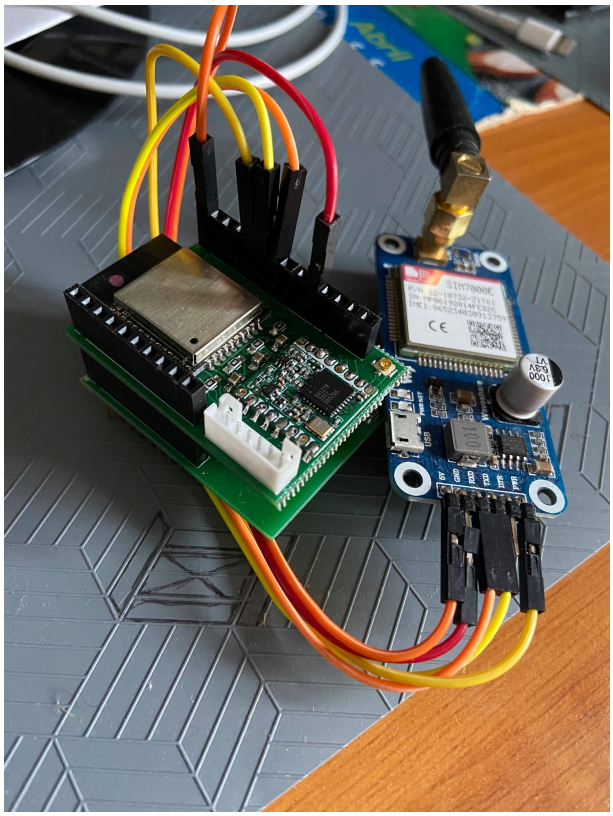


FIGURE 6.6. Leak Detection Modular Aggregation Node

For the Sensor Node, the Sensor Mode, described in Section 5.2.3.2, was used. Composed by the Main Core, the Battery Module and the Sensor Module, is capable of retrieving the water flow sensor values and sending them, via LoRa, to the aggregation node. Figure 6.7 shows the modular sensor node.

The best configuration for the Random Forest Classifier model, using Cloud Computing, was with the following parameters:

- n_estimators – 158
- split_criteria – 'gini'
- max_features – 'auto'
- max_depth – 138
- min_samples_split – 5
- min_samples_leaf – 2
- bootstrap – True

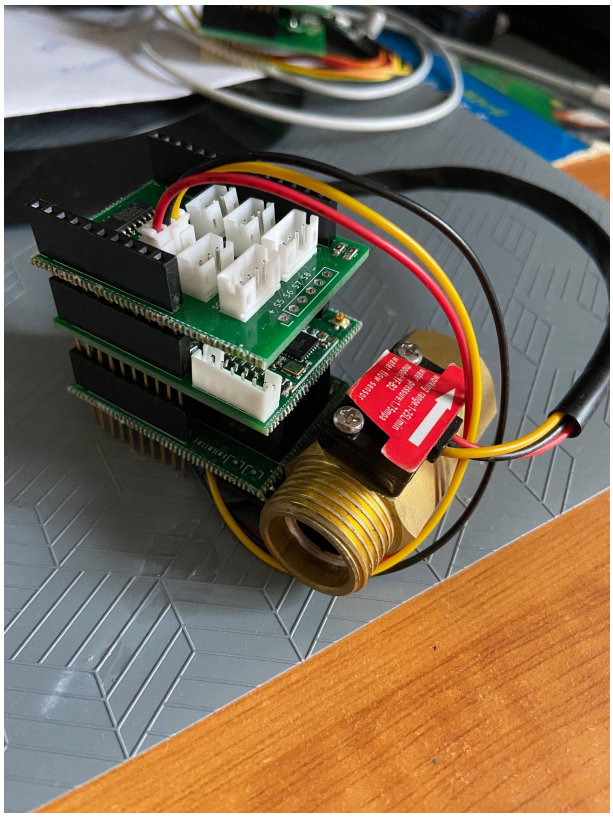


FIGURE 6.7. Leak Detection Modular Sensor Node

This model achieved an 84.79% accuracy and a file size, after porting, of 126 MB. As discussed in Section 5.3.4, in order to fit our modular approach, the model needs to be under 600 kB. So, using the methodology described in Section 3.3, the number of estimators and the depth were adjusted. Table 6.1 shows the ported model characteristics facing the trained cloud model.

TABLE 6.1. Leak Detection System Edge Computing Model Characteristics

Model	**Accuracy [%]**	**File Size [MB]**	**Estimators**	**Depth**
Cloud Computing	84.79	126	158	138
Edge Computing	80.13	0.238	20	10

By reducing 87% the number of estimators and 93% the model depth, it was possible to reach a model with 238 kB, 81% smaller than the original model, while decreasing less than 4% in accuracy. This edge computing model was implemented in the aggregation node, as the sensor node does not have the other sensor nodes information needed to run the model.

After creating a similar system using our sustainable modular approach, the exact same simulation used in the traditional approach was implemented. Once again, the

system was left running for several hours, collecting the water flow values and sending them to the aggregation node, being this time analysed directly in the node, instead of in the server. New pipes were added into the system, and as in the previous scenario, over time holes were made to simulate ruptures.

Figure 6.8 shows the obtained results, once again allowing us to see when true positives, true negatives, false positives, and false negatives were obtained for each of the possible outputs. With this, it is possible to understand the situations where the outputs were wrongly predicted and to calculate the accuracy of the system for that particular test.

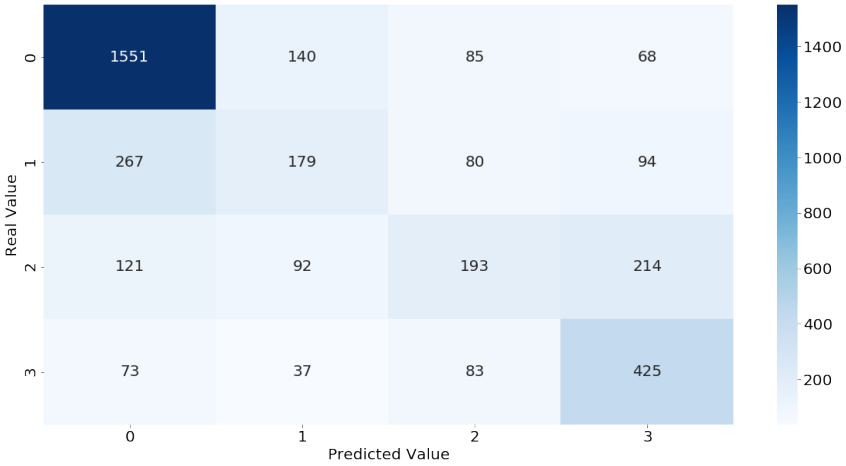


FIGURE 6.8. Modular Implementation Results

The traditional approach achieved an 75% accuracy, with 2374 correct outputs, and 82 cases of major leaks identified as non-problems, and the reverse situation happening 87 times. With our new approach, we got a similar result, achieving an 73% accuracy, with 2348 correct outputs, 2% lower. This can be justified by the lower accuracy from the edge computing model. Even so, the problems detected in the traditional approach occur less times, with 73 major leaks being detected as non-problems and the reverse situation occurring 68 times.

This proves that our solution can work as well as a traditional IoT solution.

6.2.3. Comparison

Since our system is supposed to be a more sustainable approach, not only the quality of the service needs to be compared. With the implementation of both approaches under the same scenario, it was possible to compare its accuracy, decision time, energy used and messages exchanged. Table 6.2 shows the obtained results for both scenarios.

Although with a lower accuracy, our new approach was able to decrease the energy consumption in 31%, not only due to the low power modules used but also since less

TABLE 6.2. Leak Detection System Results Comparison

Scenario	Accuracy [%]	Average Decision Time [μs]	Average Energy Used [mA]	Average Messages Exchanged	
				LoRa	NB-IoT
Cloud Computing	75	736287	136	3703	3703
Edge Computing	73	14283	94	3703	1690

messages were sent to the server for analysis, being this done directly on the aggregation node. In terms of latency, our new approach is over 50 times faster to get a decision, when facing cloud computing. Finally, in terms of message exchange, a message was only sent to the server when a problem was encountered, allowing for a reduction of 2013 NB-IoT messages.

With these results it is possible to conclude that our sustainable modular approach is indeed more sustainable than a traditional IoT approach, not only reducing the power consumption of the entire system but also reducing the costs of data transmission.

6.3. Water Management for Sustainable Farming Irrigation

In order to improve the sustainability and efficiency of irrigation in agricultural fields, a system was developed using a WSN, artificial intelligence and Machine Learning, that work according to the field needs being autonomously adapted. Through the collection of real time local data directly in the field, it is possible to improve the irrigation system, as well as give the owner the appropriate information, such as the best time of the day for irrigation. This system allows the farmer to have a better understanding of their fields and reduce the costs of water and maintenance.

6.3.1. Traditional IoT Approach

The system uses a wide range of sensors that are strategically spread over the agricultural fields in order to collect the data needed for the correct monitoring, using a WSN [**128**]. The system needs to include several Sensor Nodes, for data collection, an Actuator Node, to turn *ON/OFF* the irrigation system, and an Aggregation Node, to manage the network and send/receive messages from the server.

The aggregation node is the same from the previous case study, Figure 6.2, composed by an ESP32-DevKitC, a RFM95W, for LoRa transmissions between the nodes, and a SIM7000E NB-IoT module for communication with the server via MQTT.

The sensor node, Figure 6.9, also as the previous case study, is composed of an ESP32-DevKitC and a RFM95W, for transmitting the gathered data over LoRa, and a set of sensors. In this case, it was complemented with a SI7021, a temperature & humidity sensor, a DS18B20, a waterproof soil temperature sensor, and an analog capacity soil moisture sensor, a humidity waterproof sensor which provides the capacity of collecting data with high precision. In this scenario, the sensor node was powered by a 31 cm^2 solar panel and a 3.7V 4000mAh LiPo battery, which can last 235 days without solar exposure, with and average consumption of 42.58mA.

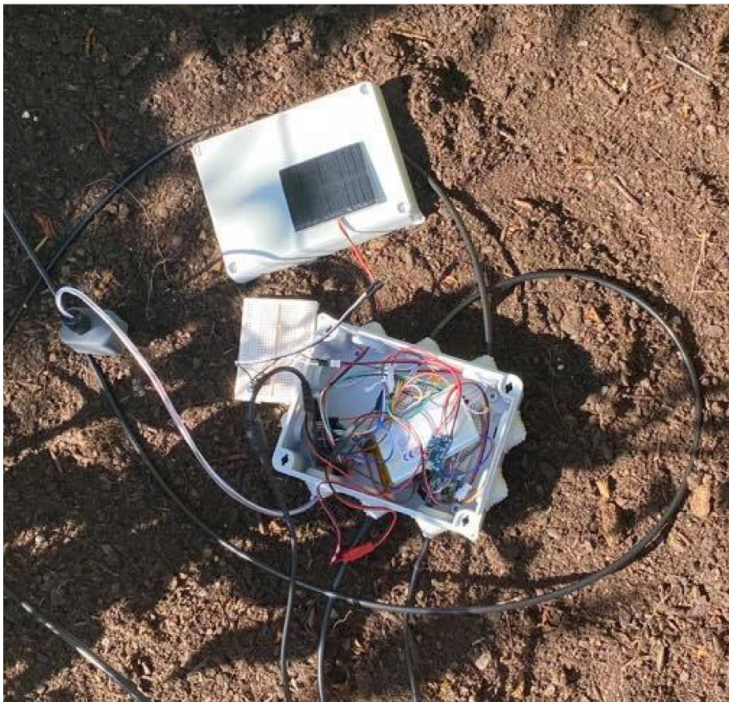


FIGURE 6.9. Irrigation Management Traditional Sensor Node [**128**]

As for the actuator node, Figure 6.10, it shares the same core of the other nodes, with an ESP32-DevKitC and a RFM95W LoRa module. It is also connected to a weather station, the SEN 0186 [**129**], capable of collecting measured wind speed, wind direction and precipitation values, that need to be constantly collecting data, thus not being able to work with battery powered sensor nodes. To be able to control the 24V irrigation pumps the Panasonic AQY212EHAT Solid State Relays (SSR) were used, being this node powered by a 24V transformer.

In the server, the collected sensor data was analyzed and transformed into knowledge to understand the real amount of water needed or the best time of day to irrigate. For that, two different approaches were done, one only based on calculations using the sensor data, to discover the optimal irrigation time, and a second one based on a Machine Learning

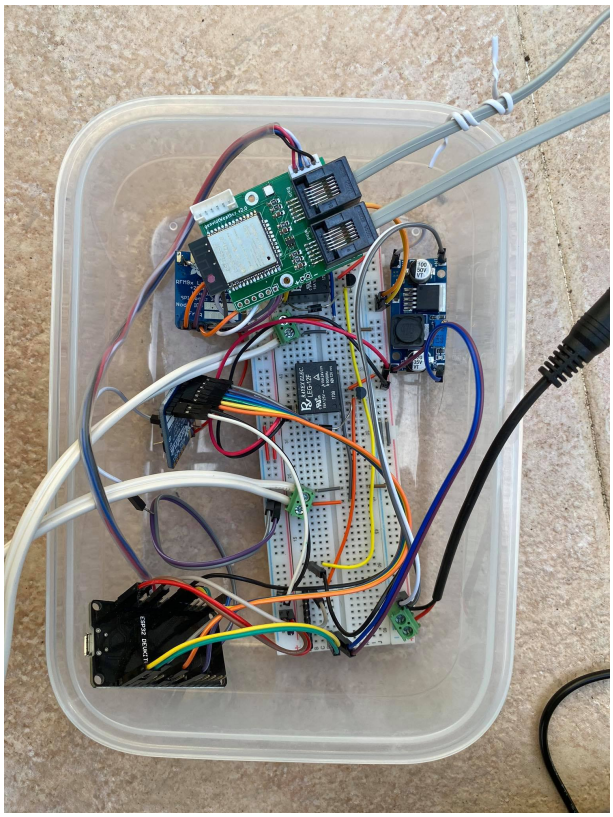


FIGURE 6.10. Irrigation Management Traditional Actuator Node [**128**]

approach, using the sensor data to predict the best irrigation hour and then calculate the irrigation time.

The calculation approach is described in detail in [**128, 130**] and take into account the use of soil moisture, air temperature and humidity, rain and wind sensors and considering the type of crops, the type of valves and tubing used, the distance between these same valves the number of irrigation in one day and the evapotranspiration.

The Machine Learning approach uses a Random Forest Classifier, that based on the collected data from the sensor nodes in the field, is capable of predicting the best irrigation hour with an 85% accuracy. Once again, this classification method was chosen based on a previous study done in [**128**].

The system was implemented in a small urban farm, whose field management was performed by the owner, who watered it once a day, around 20:00, using a hose, being the only days that it was not irrigated, the rainy ones. Figure 6.11 shows the urban farm used for the test.

The managed field was divided in two irrigation zones, with each one assigned the following sensors: air temperature and humidity, and soil moisture and temperature sensors.

FIGURE 6.11. Tested Urban Farm [**128**]

Besides that, the data of wind speed, wind direction and the rainfall were also collected by the weather station. The sensor implementation on the field can be seen in Figure 6.12.

FIGURE 6.12. System Implementation—Sensor Nodes [**128**]

Regarding irrigation, it was installed one Actuator Node that managed two water pumps, each one attached to each irrigation zone, as seen in Figure 6.13.

In the first zone was implemented the calculation approach, which verifies every hour if there are some zones to be irrigated, in case of matching, the script calculates the amount of water needed for that zone. Since the owner usually irrigates his garden once

FIGURE 6.13. System Implementation—Actuator Node [**128**]

a day, with the exception of rainy days, at the same hour, it was programmed that the irrigation zone that this algorithm was taking care of would be irrigated at 20:00, every day, according to the algorithm decisions.

In the second zone, the calculation approach was complemented with the Machine Learning approach. By implementing these two algorithms in parallel, the field was autonomous, since neither the irrigation hour nor the irrigation time had to be entered manually into the system.

To expose the system to different situations, the test was in operation for three months, between September and November of 2020, where it was exposed to good weather and rainy days.

Table 6.3 presents the results for the entire test, for the three months, based on water used to irrigate the field in each exposed situation, where the third method is the one used by the owner to irrigate his garden.

TABLE 6.3. Traditional Approach Water Usage

Test	**Days**	**Avg. Water Used per Day [L]**			**Total Consumption [L]**		
		1	**2**	**3**	**1**	**2**	**3**
Sunny	45	3.12	2.33	5.75	140.4	104.85	258.75
Rainy	28	0	0	0	0	0	0
After Rain	17	2.17	2.22	5.75	36.89	37.74	97.75
Total	90				177.29	142.59	356.5

The calculation approach is able to achieve about 50% of savings, when compared with traditional methods, in this case manual irrigation. When combining it with a Machine Learning approach that predicts the best irrigation hour, it is possible to save up to 60% more water than traditional methods, with more than 214 L of water saved in only 90 days.

6.3.2. Sustainable Modular Approach

As in the previous case study, the same system architecture from the traditional approach was followed, being the nodes replaced by our sustainable modular approach and the Machine Learning model implemented in an Edge Computing model, directly in the nodes, instead of the Cloud Computing model in the server.

For the Aggregation Node, the one from the leak detection system was used, as shown in Figure 6.6.

For the Sensor Node, the Sensor Mode, described in Section 5.2.3.2, was used. Composed by the Main Core, the Solar Module and the Sensor Module, is capable of retrieving the sensor values and sending them, via LoRa, to the aggregation node. To the Solar Module, the same 31 cm^2 solar panel and 3.7V 4000mAh LiPo battery from the traditional approach, were connected. Figure 6.14 shows the modular sensor node.

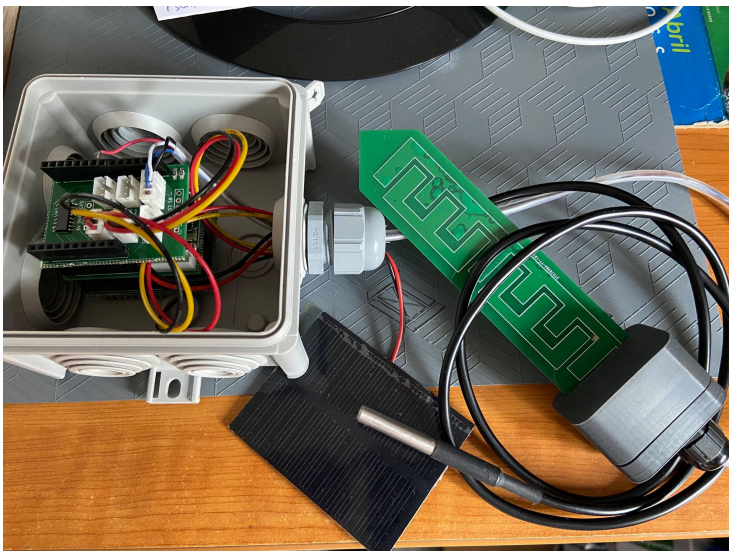


FIGURE 6.14. Irrigation Management Modular Sensor Node

For the Actuator Node, the Controller Mode, described in Section 5.2.3.3, was used. Using the Main Core, the DC Power Module and the DC Actuator Module, it was possible to control the 24V irrigation pumps and power the node over the 24V power transformer.

Also the Sensor Module was attached to connect the weather station sensors. Figure 6.15 shows the modular sensor node.

FIGURE 6.15. Irrigation Management Modular Actuator Node

The best configuration for the Random Forest Classifier model, using Cloud Computing, was with the following parameters:

- n_estimators – 212
- split_criteria – 'gini'
- max_features – 'auto'
- max_depth – 196
- min_samples_split – 10
- min_samples_leaf – 2
- bootstrap – True

This model achieved an 85.49% accuracy and a file size, after porting, of 559 MB. As discussed in Section 5.3.4, in order to fit our modular approach, the model needs to be under 600 kB. So, using the methodology described in Section 3.3, the number of estimators and the depth were adjusted. Table 6.4 shows the ported model characteristics versus the trained cloud model.

TABLE 6.4. Irrigation Management System Edge Computing Model Characteristics

Model	**Accuracy [%]**	**File Size [MB]**	**Estimators**	**Depth**
Cloud Computing	85.49	556	212	196
Edge Computing	80.17	0.192	25	5

By reducing 89% the number of estimators and 98% the model depth, it was possible to reach a model with 192 kB, 99.97% smaller than the original model, while decreasing less than 6% in accuracy. This edge computing model was implemented in the sensor node, that after getting a decision send a LoRa message to the actuator node to start the irrigation process.

After creating a similar system using our sustainable modular approach it was implemented in the same urban farm as in the previous test. This time only one zone was controlled, the zone number 2 from the previous test, and the Machine Learning coupled with the calculation approach method was used, since it proved to be the best solution. To, once again, expose the system to different situations, the test was in operation for three months, between March and May of 2021, where it was exposed to good weather and rainy days.

Table 6.5 presents the results for the entire test, for the three months, based on water used to irrigate the field in each exposed situation, where the third method is the one used by the owner to irrigate his garden.

TABLE 6.5. Sustainable Modular Approach Water Usage

Test	**Days**	**Avg. Water Used per Day [L]**		**Total Consumption [L]**	
		2	**3**	**2**	**3**
Sunny	60	2.36	5.75	141.60	345.00
Rainy	18	0	0	0	0
After Rain	12	2.28	5.75	27.36	60.00
Total	90			168.96	405.00

Our new sustainable and modular approach, with the edge computing decision system, achieved the same 60% water savings over the manual irrigation, with 236 L, or 52 L/m^2, of water saved. Since the actuation process in this system does not depend heavily on Machine Learning, it is the calculation process that decides the amount of water to use,

it is normal not to find many differences in the efficiency and quality of the service when compared to the traditional IoT approach.

6.3.3. Comparison

Once again, since our system is supposed to be a more sustainable approach, not only the quality of the service needs to be compared. With the implementation of both approaches under the same scenario, it was possible to compare its accuracy, decision time, energy used and messages exchanged. Table 6.6 shows the obtained results for both scenarios.

TABLE 6.6. Irrigation Management System Results Comparison

Scenario	**Accuracy** [%]	**Average Decision Time** [μs]	**Average Energy Used** [mA]	**Average Messages Exchanged**	
				LoRa	**NB-IoT**
Cloud Computing	85.49	589274	159	2201	2201
Edge Computing	80.17	8737	86	25	0

Although with a lower accuracy, our new approach was able to decrease the energy consumption in 46%, not only due to the low power modules used but also since less messages were sent to the server for analysis, being this done directly on the sensor node. Compared to the previous scenario, even less energy was used since no LoRa messages were needed to send the data to the aggregation node to be analyzed, since it was done directly on the sensor node, using 2176 less LoRa messages. With this, also no NB-IoT messages were needed for the decision process nor the actuation on the environment. In terms of latency, the edge computing approach is 67 times faster.

These results further validate that our sustainable modular approach is capable of replacing a traditional approach while being more sustainable, reducing the power consumption of the entire system but also reducing the costs of data transmission and decision time.

6.4. Remarks

This chapter presented the implementation of the new developed system, based on the research and development presented in the previous chapters. It presented a detailed approach on using sustainable modular IoT solutions to replace traditional IoT systems, in order to create more sustainable processes and solutions.

With the goal of assessing if the new solution would have a similar performance to other IoT solutions already developed and tested in real case scenarios, two previous research projects, both in the field of water management in Smart Cities, were selected to be replaced with this new approach.

In the first one, used to detect leaks in water supply pipes for houses or irrigation systems, the system used a set of sensor nodes, combined with an aggregation node, to send the sensor data to the cloud to be analyzed. With an accuracy of 75%, it proved to be a reliable solution to detect leaks. Using the new solution, we replaced the entire hardware with the smart nodes, which in a modular way were able to create the sensor and aggregation nodes. The data analysis model was ported into an edge computing model and inserted in the aggregation node. Although losing some accuracy, being 2% lower, this allowed for the reduction of more than 2013 exchanged messages, creating a more power saving system, able to reduce up to 31% the power consumption of the overall system.

In the second scenario, used to reduce the water consumption of irrigation systems, the system used sensor nodes, to collect the field conditions using sensors, an actuator node, to control the irrigation pumps, and an aggregation node, to send the data to the cloud and receive the irrigation configuration timings. This system, which used Machine Learning to predict the best irrigation hour and a set of calculations to predict the water needs for the field, was able to reduce up to 60% of the water consumption comparing to the manual irrigation. With the new approach, we switched all the hardware to our modular smart nodes and ported the Machine Learning model to run directly on the sensor node. Although the same 60% water savings were achieved with the new approach, it was done using 46% less energy, since there was a reduction of 2176 LoRa messages and 2201 NB-IoT messages. Also, a 67 times faster decision process was obtained.

These two implementation scenarios showed that not only this new solution can be applied to different systems, using different nodes under specific configurations, but that can also do it in a more sustainable and efficient way. It also proves that the new solution is ready for the market and that it can easily undergo a technology transfer process.

With the results presented in this chapter, the entire research is validated and proven to be useful to create a sustainable modular IoT solution for Smart Cities or regions applications supported by Machine Learning.

CHAPTER 7

Conclusions

With the proliferation of Smart Cities, supported by Internet of Things, Wireless Sensor Networks, Wireless Communications and Machine Learning, a new set of challenges rise and put in question the sustainability and security of these solutions. New research and product development must focus on creating green devices that not only reduce the energy consumption, but also create more efficient and secure communications, storage and data mining, while reducing the complexity of deployment, inter-connectivity and scalability. It is only after these problems are no longer a reality that the resistance and mistrust in Smart Cities will disappear and the population start to accept them.

This thesis presented a new solution for Smart Cities applications based on modular IoT smart nodes supported by Machine Learning that can create more sustainable and efficient systems. This solution is composed of smart nodes that can self-configure its software, communication configuration and data analysis, without human intervention, allowing for an easy deployment into new services. For that a set of modules were developed that, when connected, will create an autonomous solution capable of adapting to any situation or configuration, in order to achieve maximum efficiency in the associated task, but also do it in a more sustainable way, reducing the decision times, power consumption and costs. These modules include the smart node, a fully adaptable and modular node that can create WSN in a one-fits-all solution; an autonomous communication configuration system, capable of tackling both device-to-device and device-to-server communications, using the best protocol available and its configuration, based on location and conditions; and a set of learning systems, based on Machine Learning, that not only are the basis of both of the previous modules, but will also help analyse the data and extract the need knowledge to improve efficiency and the performance of the system.

The learning systems were the first module to be presented, as they were the basis for the entire solution. Several algorithms were tested with the goal of assessing if one solution could support the entire system or if multiple solutions were needed based on the system needs. Decision Trees, Random Forest, Neural Networks, Linear Regressions and Support Vector Machines were tested under different scenarios and datasets and Random

Forest proved to be the best solution in all of them, for both classifications and regressions. Although this happens, it is possible to conclude that it is important to analyse several solutions for the same problem, as each could benefit from a specific configuration. The next step was to understand if running the learning system directly on the edge devices could improve the system performance and energy efficiency. For that, a study was done on how the fully trained cloud computing models could be ported into the low-memory microcontrollers and how it affects its performance. Although it reduces the accuracy of the models, edge computing proved to also reduce latency and power consumption, being 90% faster using 50% less power with an accuracy only 6% lower. As such it is possible to conclude that more sustainable decision processes can be achieved without compromising its efficiency and reliability.

The next phase was to apply these learning systems to create an autonomous communication configuration system, capable of creating a more sustainable message exchange system, for device-to-device and device-to-server, based on the available protocols, node location and environment conditions. For device-to-device, it started with understanding how the most typical protocol behaves in different scenarios, such as indoor or outdoor, rural or urban and with or without obstacles. ESPNow, Bluetooth Low Energy, FSK Radio Frequency, LoRa and ZigBee were tested and, although using a higher power consumption, LoRa proved to be the best overall protocol, as it was the only one able to adapt to the indoor and outdoor environment on both short and long distances and with or without obstacles. It was also possible to conclude that using a higher transmission power does not always create the best communication link, mainly when tackling shorter distances. Knowing how each protocol behaves, it was possible to create a configuration system for point-to-point communication that predicts the link quality and energy consumption of each protocol and configuration and choose the best one based on quality efficiency, energy efficiency or an hybrid approach. With this, it was possible to create a communication system capable of saving 65% of the energy needed to send a message while only reducing 13% the quality of the network. The same methodology was applied to the cloud communication system, for using WiFi, Cellular, LoRaWAN or SigFox. In this scenario, edge computing proved not to be able to cope with the task, not only having a higher margin of error and a higher decision time, it was not able to use the full methodology due to the lack of memory of the devices. Nevertheless, the cloud computing

model proved to be able to configure the node with the best available protocol based on the nearby base stations and protocols available.

The final developed module was the smart IoT node, a modular node capable of acting as a gateway, sensor or actuator node and that is self-configurable based on the attached modules. For this self-configuration, a set of edge computing models were used to discover the attached modules, if they were compatible and then to configure the node tasks based on the attached modules. These models proved to work, with an accuracy of 89%, on the worst scenario, being able to create a fully adaptable and autonomous node.

With the goal of assessing if the developed nodes could be put together to create a new solution capable of providing a more efficient and sustainable system, it was compared with two typical IoT solutions in a Smart City environment: leak detection in the water distribution pipelines and a smart irrigation system for urban farms. For each of these scenarios, this new solution was assembled in order to mimic the smart solution already implemented and tested, using multiple smart nodes allocated to the specific tasks. In the first scenario, several sensor nodes were used to collect the water flow information and an edge computing model was implemented in the gateway to analyse those data. After testing this new solution versus the previous implementation, it was possible to conclude that this solution, although with an accuracy 2% lower, was able to reduce the power consumption of the entire system by 31% while also reducing the number of exchanges messages, creating not only a more sustainable system but also increasing the quality of the network. In the second scenario, several sensor nodes were used to collect information about the urban farm fields conditions and an actuator node to control the irrigation pumps. With these, it was possible to reduce up to 60% the consumption of water, a similar result facing the previous smart solution, but with 46% less power needed and with almost no messages exchanged.

These results prove that our solution not only is capable of having the same performance as a typical IoT solution but will do it in a more sustainable and efficient way, allowing for Smart Cities to reach its full potential and solving some of the challenges underlying its fast proliferation.

Although a full solution is provided, alongside all its research and methodologies, that is ready for a technology transfer process and to be inserted into to the market, still some improvements and new research can be done, mainly:

- Battery Optimization: Create mechanisms to improve the battery usage;

- Cost Reduction: Revised the used hardware to create cheaper nodes;
- Deep Learning: Update the learning system to use more recent Machine Learning methods, such as Deep Learning, in order to improve overall accuracy and performance.

Beside these, as stated in Chapter 2, there are still several challenges that were not tackled in this thesis, such as security, storage or data trust, that still can be researched and where this innovative solution can also help. All of these can be further developed in the future.

References

[1] S. K. Routray and K. P. Sharmila, "Green initiatives in IoT," in *Proceedings of the 3rd IEEE International Conference on Advances in Electrical and Electronics, Information, Communication and Bio-Informatics, AEEICB 2017*. Institute of Electrical and Electronics Engineers Inc., 7 2017, pp. 454–457.

[2] M. Nasiri, N. Tura, and V. Ojanen, "Developing Disruptive Innovations for Sustainability: A Review on Impact of Internet of Things (IOT)," *2017 Proceedings of PICMET '17: Technology Management for Interconnected World*, 2017.

[3] J. Chin, V. Callaghan, and I. Lam, "Understanding and personalising smart city services using machine learning, the Internet-of-Things and Big Data," in *IEEE International Symposium on Industrial Electronics*. Institute of Electrical and Electronics Engineers Inc., 8 2017, pp. 2050–2055.

[4] T. S. Brisimi, C. G. Cassandras, C. Osgood, I. C. Paschalidis, and Y. Zhang, "Sensing and Classifying Roadway Obstacles in Smart Cities: The Street Bump System," *IEEE Access*, vol. 4, pp. 1301–1312, 2016.

[5] E. Tabane, S. M. Ngwira, and T. Zuva, "Survey of smart city initiatives towards urbanization," in *Proceedings - 2016 3rd International Conference on Advances in Computing, Communication and Engineering, ICACCE 2016*. Institute of Electrical and Electronics Engineers Inc., 10 2017, pp. 437–440.

[6] J. Nigon, E. Glize, D. Dupas, F. Crasnier, and J. Boes, "Use Cases of Pervasive Artificial Intelligence for Smart Cities Challenges," in *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress*, 2016, pp. 1021–1027.

[7] J. Kane, B. Tang, Z. Chen, J. Yan, T. Wei, H. He, and Q. Yang, "Reflex-tree: A biologically inspired parallel architecture for future smart cities," in *Proceedings of the International Conference on Parallel Processing*, vol. 2015-December. Institute of Electrical and Electronics Engineers Inc., 12 2015, pp. 360–369.

[8] K. Mohan and L. Cao, "Managing Disruptive and Sustaining Innovations in Green IT," *IT Professional*, pp. 22–29, 2012.

[9] M. A. M Albreem, A. A. El-Saleh, M. Isa, W. Salah, M. Jusoh, M. Azizan, A. Ali, and M. Perlis, "Green Internet of Things (IoT): An Overview," *Proc. of the 4th IEEE International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA)*, 2017.

[10] M. Bergés and C. Samaras, "A path forward for smart cities and iot devices," *IEEE Internet of Things Magazine*, vol. 2, pp. 2–4, 10 2019.

[11] L. Atzori, A. Iera, G. Morabito, and A. Diee, "The Internet of Things : A survey," *Computer Networksxxx*, 2010.

[12] D. Evans, "The Internet of Things How the Next Evolution of the Internet Is Changing Everything," *CISCO, San Jose, CA, USA, White Paper*, 2011.

[13] D. Lee, D. Lough, S. Midkiff, N. Davis, and P. Benchoff, "The next generation of the Internet: aspects of the Internet protocol version 6," *IEEE Network*, vol. 12, no. 1, 1998.

[14] 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. IEEE, 2018, pp. 1156–1161.

[15] J. A. Stankovic, "Research directions for the internet of things," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 3–9, 2014.

[16] 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*, 2015.

[17] F. Wortmann and K. Flüchter, "Internet of Things: Technology and Value Added," *Business and Information Systems Engineering*, vol. 57, no. 3, pp. 221–224, 6 2015.

[18] R. R. Reddy, C. Mamatha, and R. G. Reddy, "A Review on Machine Learning Trends, Application and Challenges in Internet of Things," in *2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2018*. Institute of Electrical and Electronics Engineers Inc., 11 2018, pp. 2389–2397.

[19] R. Kitchin, "The real-time city? Big data and smart urbanism," *GeoJournal*, vol. 79, no. 1, pp. 1–14, 2014. [Online]. Available: https://doi.org/10.1007/s10708-013-9516-8

[20] C. Kyriazopoulou, "Smart city technologies and architectures: A literature review," in *2015 International Conference on Smart Cities and Green ICT Systems (SMARTGREENS)*, 2015, pp. 1–12.

[21] IBM, "Smarter Planet," 2010. [Online]. Available: https://www.ibm.com/ibm/history/ibm100/us/en/icons/smarterplanet/

[22] K. Su, J. Li, and H. Fu, "Smart city and the applications," in *2011 International Conference on Electronics, Communications and Control, ICECC 2011 - Proceedings*, 2011, pp. 1028–1031.

[23] A. Cocchia, "Smart and Digital City: A Systematic Literature Review," in *Smart City. Progress in IS.*, Dameri R. and Rosenthal-Sabroux C., Eds. Springer, Cham, 6 2014, pp. 13–43.

[24] M. Abu-Matar, "Towards a software defined reference architecture for smart city ecosystems," in *2016 IEEE International Smart Cities Conference (ISC2)*, 2016, pp. 1–6.

[25] M. Mohammadi and A. Al-Fuqaha, "Enabling Cognitive Smart Cities Using Big Data and Machine Learning: Approaches and Challenges," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 94–101, 2 2018.

[26] H. Habibzadeh, A. Boggio-Dandry, Z. Qin, T. Soyata, B. Kantarci, and H. T. Mouftah, "Soft Sensing in Smart Cities: Handling 3Vs Using Recommender Systems, Machine Intelligence, and Data Analytics," *IEEE Communications Magazine*, vol. 56, no. 2, pp. 78–86, 2 2018.

[27] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2 2014.

[28] M. Abdur Rahman, M. Shamim Hossain, E. Hassanain, and G. Muhammad, "Semantic Multimedia Fog Computing and IoT Environment: Sustainability Perspective," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 80–87, 5 2018.

[29] M. Pouryazdan, B. Kantarci, T. Soyata, L. Foschini, and H. Song, "Quantifying user reputation scores, data trustworthiness, and user incentives in mobile crowd-sensing," *IEEE Access*, vol. 5, pp. 1382–1397, 2017.

[30] J. Canedo and A. Skjellum, "Using machine learning to secure IoT systems," in *2016 14th Annual Conference on Privacy, Security and Trust, PST 2016*. Institute of Electrical and Electronics Engineers Inc., 2016, pp. 219–222.

[31] M. Usmonov and F. Gregoretti, "Design and Implementation of a LoRa Based Wireless Control for Drip Irrigation Systems," *2nd International Conference on Robotics and Automation Engineering*, pp. 248–253, 2017.

[32] G. E. John, "A Low Cost Wireless Sensor Network for Precision Agriculture," *Sixth International Symposium on Embedded Computing and System Design (ISED)*, pp. 24–27, 2016.

[33] S. Ray, "A Quick Review of Machine Learning Algorithms," in *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)*, 2019.

[34] S. Bhattacharya, S. R. K. Somayaji, T. R. Gadekallu, M. Alazab, and P. K. R. Maddikunta, "A review on deep learning for future smart cities," *Internet Technology Letters*, pp. 1–6, 6 2020.

[35] I. U. Din, M. Guizani, J. J. Rodrigues, S. Hassan, and V. V. Korotaev, "Machine learning in the internet of things: Designed techniques for smart cities," *Future Generation Computer Systems*, vol. 100, pp. 826–843, 11 2019.

[36] S. Murugeswari, D. R. Ravi, and S. Raji, "Comparison Between WSN and WSAN," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, vol. 3, no. 4, pp. 1267–1272, 2014.

[37] R. Shuker and J. Taylor, "Context-aware networking and communications: part 2," *IEEE Communications Magazine*, vol. 39, no. 1, pp. 64–65, 2014.

[38] D. Martín, C. Lamsfus, and A. Alzua, "Automatic context data life cycle management framework," in *ICPCA10 - 5th International Conference on Pervasive Computing and Applications*, 2010, pp. 330–335.

[39] Z. Yan, X. Yu, and W. Ding, "Context-aware verifiable cloud computing," *IEEE Access*, vol. 5, pp. 2211–2227, 2017.

[40] Q. H. Cao, I. Khan, R. Farahbakhsh, G. Madhusudan, G. M. Lee, and N. Crespi, "A trust model for data sharing in smart cities," in *2016 IEEE International Conference on Communications, ICC 2016*. Institute of Electrical and Electronics Engineers Inc., 7 2016.

[41] Y. Han, Y. Chen, B. Wang, and R. K. J. Liu, "Enabling Heterogeneous Connectivity in Internet of Things: A Time-Reversal Approach," *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 1036–1047, 12 2016.

[42] A. Meddeb, "Internet of things standards: who stands out from the crowd?" *IEEE Communications Magazine*, vol. 54, no. 7, pp. 40–47, 2016.

[43] M. Ali, S. U. Khan, and A. Y. Zomaya, "Security and Dependability of Cloud-Assisted Internet of Things," *IEEE Cloud Computing*, vol. 3, no. 2, pp. 24–26, 5 2016.

[44] Y. E. Song, Y. Liu, S. Fang, and S. Zhang, "Research on applications of the internet of things in the smart grid," in *Proceedings - 2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics, IHMSC 2015*, vol. 2. Institute of Electrical and Electronics Engineers Inc., 11 2015, pp. 178–181.

[45] Q. Meng and J. Jin, "The terminal design of the energy self-sufficiency Internet of Things," in *2011 International Conference on Control, Automation and Systems Engineering, CASE 2011*, 2011.

[46] Keysight Technologies, "Battery Life Challenges in IoT Wireless Sensors and the Implications for Test," Tech. Rep., 2015.

[47] S. Tozlu, M. Senel, W. Mao, and A. Keshavarzian, "Wi-Fi Enabled Sensors for Internet of Things: A Practical Approach," *IEEE Communications Magazine*, vol. 50, no. 6, pp. 134–143, 2012.

[48] B. Ahlgren, M. Hidell, and E. C. Ngai, "Internet of Things for Smart Cities: Interoperability and Open Data," *IEEE Internet Computing*, vol. 20, no. 6, pp. 52–56, 11 2016.

[49] E. Ismagilova, L. Hughes, N. P. Rana, and Y. K. Dwivedi, "Security, privacy and risks within smart cities: Literature review and development of a smart city interaction framework," *Information Systems Frontiers*, 2020.

[50] 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*. Institute of Electrical and Electronics Engineers Inc., 9 2018, pp. 215–218.

[51] S. Murugesan, "Harnessing Green IT: Principles and Practices," *IEEE IT Prof.*, vol. 10, no. 1, pp. 24–33, 2008. [Online]. Available: http://egj.lib.uidaho.edu/index.

[52] X. Fafoutis, L. Marchegiani, A. Elsts, J. Pope, R. Piechocki, and I. Craddock, "Extending the battery lifetime of wearable sensors with embedded machine learning," in *IEEE World Forum on Internet of Things, WF-IoT 2018 - Proceedings*, vol. 2018-January. Institute of Electrical and Electronics Engineers Inc., 5 2018, pp. 269–274.

[53] D. Thomas, R. Mcpherson, G. Paul, J. Irvine, and . Keith Bishop, "Optimizing Power Consumption of Wi-Fi for IoT Devices An MSP430 processor and an ESP-03 chip provide a power-efficient solution," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 92–100, 2016.

[54] Z. Dawy, W. Saad, A. Ghosh, J. G. Andrews, and E. Yaacoub, "Toward Massive Machine Type Cellular Communications," *IEEE Wireless Communications*, vol. 24, no. 1, pp. 120–128, 2 2017.

[55] V. M. Suresh, R. Sidhu, P. Karkare, A. Patil, Z. Lei, and A. Basu, "Powering the IoT through embedded machine learning and LoRa," in *IEEE World Forum on Internet of Things, WF-IoT 2018 - Proceedings*, vol. 2018-January. Institute of Electrical and Electronics Engineers Inc., 5 2018, pp. 349–354.

[56] V. Kumar, R. Jafari, and S. Pourkamali, "Ultra-Low Power Digitally Operated Tunable MEMS Accelerometer," *IEEE Sensors Journal*, vol. 16, no. 24, pp. 8715–8721, 12 2016.

[57] N. A. Pantazis, S. A. Nikolidakis, and D. D. Vergados, "Energy-efficient routing protocols in wireless sensor networks: A survey," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 2, pp. 551–591, 2013.

[58] A. Di Mauro, X. Fafoutis, S. Mödersheim, and N. Dragoni, "Detecting and preventing beacon replay attacks in receiver-initiated MAC protocols for energy efficient WSNs," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8208 LNCS. Springer Verlag, 2013, pp. 1–16.

[59] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *29th Annual IEEE International Conference on Local Computer Networks*, 2004, pp. 455–462.

[60] J. Park, H. Park, and Y. J. Choi, "Data compression and prediction using machine learning for industrial IoT," in *International Conference on Information Networking*, vol. 2018-January. IEEE Computer Society, 4 2018, pp. 818–820.

[61] Green Power for Mobile, "The Global Telecom Tower ESCO Market," Technical Report, Tech. Rep., 2014.

[62] A. Fehske, G. Fettweis, J. Malmodin, and G. Biczók, "The Global Footprint of Mobile Communications: The Ecological and Economic Perspective," *IEEE Communication Magazine*, vol. 49, no. 8, pp. 55–62, 2011.

[63] U. N. D. of Economic and S. Affairs, "Sustainable Development Goals," 2021, [Online] Available: https://sdgs.un.org/goals, (visited 26/06/2021).

[64] ——, "Sustainable Development Goals - 11," 2021, [Online] Available: https://sdgs.un.org/goals/goal11, (visited 26/06/2021).

[65] 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, no. 4, pp. 3072–3108, 9 2018. [Online]. Available: http://arxiv.org/abs/1809.08707

[66] 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)*, 2018, pp. 945–949.

[67] Z. Ullah, F. Al-Turjman, L. Mostarda, and R. Gagliardi, "Applications of Artificial Intelligence and Machine learning in smart cities," *Computer Communications*, vol. 154, pp. 313–323, 3 2020.

[68] A. Goldstein, L. Fink, A. Meitin, S. Bohadana, O. Lutenberg, and G. Ravid, "Applying machine learning on sensor data for irrigation recommendations: revealing the agronomist's tacit knowledge," *Precision Agriculture*, vol. 19, no. 3, pp. 421–444, 6 2018.

[69] L. Breiman, "Random Forests," in *Machine Learning*, 2001, vol. 45, pp. 5–32.

[70] D. F. S. Fernandes, A. Raimundo, F. Cercas, P. J. A. 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.

[71] S. Nosratabadi, A. Mosavi, R. Keivani, S. Ardabili, and F. Aram, "State of the Art Survey of Deep Learning and Machine Learning Models for Smart Cities and Urban Sustainability," in *International Conference on Global Research and Education*, 2019, pp. 228–238.

[72] 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. IEEE Computer Society, 7 2019, pp. 916–917.

[73] 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.

[74] 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*. Institute of Electrical and Electronics Engineers Inc., 10 2017, pp. 65–68.

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

[76] J. Nalepa and M. Kawulok, "Selecting training sets for support vector machines: a review," *Artificial Intelligence Review*, vol. 52, no. 2, pp. 857–900, 8 2019.

[77] I. Sitton-Candanedo, R. S. Alonso, O. Garcia, A. B. Gil, and S. Rodriguez-Gonzalez, "A review on edge computing in smart energy by means of a systematic mapping study," *Electronics (Switzerland)*, vol. 9, no. 1, 1 2020.

[78] I. Sitton-Candanedo, R. S. Alonso, J. M. Corchado, S. Rodriguez-Gonzalez, and R. Casado-Vara, "A review of edge computing reference architectures and a new global edge proposal," *Future Generation Computer Systems*, vol. 99, pp. 278–294, 10 2019.

[79] P. R. Kumar, P. H. Raj, and P. Jelciana, "Exploring Data Security Issues and Solutions in Cloud Computing," in *Procedia Computer Science*, vol. 125. Elsevier B.V., 2018, pp. 691–697.

[80] W. Rafique, L. Qi, I. Yaqoob, M. Imran, R. U. Rasool, and W. Dou, "Complementing IoT Services through Software Defined Networking and Edge Computing: A Comprehensive Survey," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 3, pp. 1761–1804, 7 2020.

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

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

[83] A. Gloria and P. Sebastião, "Temperature Distribution Analyses with Wireless Sensor Networks and Machine Learning," in *International Conference on Sensing and Instrumentation in IoT Era (ISSI)*, 2019.

[84] J. A. Coelho, "Machine learning for precise water leaks detection," Master's thesis, ISCTE - Instituto Universitário de Lisboa, 2020.

[85] J. Cardoso, "Smartfarm: Improve sustainability using wireless sensor networks," Master's thesis, ISCTE - Instituto Universitário de Lisboa, 2020.

[86] IPMA, "IPMA - API," 2018. [Online]. Available: http://api.ipma.pt/

[87] S. Makonin, F. Popowich, L. Bartram, B. Gill, and I. V. Bajic, "AMPds: A Public Dataset for Load Disaggregation and Eco-Feedback Research," in *Electrical Power and E nergy Conference (EPEC)*, 2013, pp. 1–6. [Online]. Available: http://www.greenbuttondata.org/greendevelop.aspx

[88] 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. [Online]. Available: http://data.europa.eu/89h/jrc-netbravo-netbravo-od-eu-cellular

[89] S. Salerno, "RandomizedSearchCV," 2021, [Online] Available: https://github.com/agrimagsrl/micromlgen, (visited 03/01/2021).

[90] 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, p. 101913, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870519300812

[91] 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)*, 2018, pp. 1–6.

[92] 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.

[93] 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*. Institute of Electrical and Electronics Engineers Inc., 5 2019, pp. 908–911.

[94] 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)*, 2020, pp. 1997–2001.

[95] F. J. Dian and R. Vahidnia, "Formulation of BLE Throughput Based on Node and Link Parameters," *Canadian Journal of Electrical and Computer Engineering*, vol. 43, no. 4, pp. 261–272, 9 2020.

[96] K. E. Jeon, J. She, P. Soonsawad, and P. C. Ng, "BLE Beacons for Internet of Things Applications: Survey, Challenges, and Opportunities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 811–828, 4 2018.

[97] Espressif, "ESP-Now Overview," 3 2021.

[98] T. N. Hoang, S.-T. Van, and B. D. Nguyen, "ESP-NOWBased Decentralized Low Cost VoiceCommunication SystemsFor Buildings," in *2019 International Symposium on Electrical and Electronics Engineering (ISEE)*, 2019, pp. 108–112.

[99] A. J. Wixted, P. Kinnaird, H. Larijani, A. Tait, A. Ahmadinia, and N. Strachan, "Evaluation of LoRa and LoRaWAN for wireless sensor networks," in *Proceedings of IEEE Sensors*, 2017, pp. 1–3.

[100] A. Glória, C. Dionisio, G. Simões, and P. Sebastião, "LoRa Transmission Power Self Configuration forLow Power End Devices," in *2019 22nd International Symposium on Wireless Personal Multimedia Communications (WPMC)*, 2019.

[101] LoRa™ Alliance, "LoRaWAN™ Specification," 2015. [Online]. Available: https://www.lora-alliance.org/portals/0/specs/LoRaWANSpecification1R0.pdf

[102] A. I. Ali, S. Z. Partal, S. Kepke, and H. P. Partal, "ZigBee and LoRa based Wireless Sensors for Smart Environment and IoT Applications," in *2019 IEEE 1st Global Power, Energy and Communication Conference (GPECOM2019)*, 2019, pp. 19–23.

[103] ZigBee Alliance, "ZigBee Specifications," 2021.

[104] S. Tsantilas, C. Spandonidis, F. Giannopoulos, N. Galiatsatos, D. Karageorgiou, and C. Giordamlis, "A comparative study of wireless communication protocols in a computer vision system for improving the autonomy of the visually impaired," *Journal of Engineering Science and Technology Review*, vol. 13, pp. 72–76, 02 2020.

[105] P. J. Basford, F. M. Bulot, M. Apetroaie-Cristea, S. J. Cox, and S. J. Ossont, "Lorawan for smart city iot deployments: A long term evaluation," *Sensors (Switzerland)*, vol. 20, 2 2020.

[106] K. Mekki, E. Bajic, F. Chaxel, and F. Meyer, "A comparative study of lpwan technologies for large-scale iot deployment," *ICT Express*, vol. 5, no. 1, pp. 1–7, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405959517302953

[107] A. I. PETRARIU and A. LAVRIC, "Sigfox wireless communication enhancement for internet of things: A study," in *2021 12th International Symposium on Advanced Topics in Electrical Engineering (ATEE)*, 2021, pp. 1–4.

[108] Espressif Systems, "ESP32 Series Datasheet," 2018. [Online]. Available: www.espressif.com/en/subscribe.

[109] HopeRF, "RFM69HCW Datasheet," 2006, [Online] Available: https://cdn.sparkfun.com/datasheets/Wireless/General/RFM69HCW-V1.1.pdf, (visited 11/03/2021).

[110] ——, "RFM95/96/97/98(W) - Low Power Long Range Transceiver Module," 2019. [Online]. Available: https://cdn.sparkfun.com/assets/learn_tutorials/8/0/4/RFM95_96_97_98W.pdf

[111] Digi, "XBee 3 - RF Module Datasheet," 2021, [Online] Available: https://www.digi.com/resources/documentation/digidocs/pdfs/90001543.pdf, (visited 11/03/2021).

[112] 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. [Online]. Available: https://doi.org/10.5281/zenodo.3904158

[113] UnwiredLabs, "OpenCelliD - Largest Open Database of Celltowers & Geolocation," 2021, [Online] Available: https://www.opencellid.org, (visited 03/01/2021).

[114] RadioCell, "RadioCell - Celltowers & WiFi BaseStations Database," 2021, [Online] Available: https://radiocells.org, (visited 03/01/2021).

[115] S. Trilles, A. Gonzalez-Perez, and J. Huerta, "A comprehensive iot node proposal using open hardware. a smart farming use case to monitor vineyards," *Electronics (Switzerland)*, vol. 7, 12 2018.

[116] Atmel, "ATMega32u4 Datasheet," 2021, [Online] Available: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf, (visited 02/06/2021).

[117] ARM, "Cortex-M0+ Datasheet," 2021, [Online] Available: https://www.nxp.com/docs/en/data-sheet/LPC8N04.pdf, (visited 02/06/2021).

[118] N. Semiconductors, "nRF52840 Datasheet," 2021, [Online] Available: https://infocenter.nordicsemi.com/pdf/nRF52840_OPS_v0.5.1.pdf, (visited 02/06/2021).

[119] C. L. T. Peral, G. V. Martínez, R. P. Hernández, J. H. G. Becerril, J. G. F. Sánchez, J. G. Martínez, C. A. Serrano, A. V. Hernández, and L. L. Salas, "Experience of use of the bitalino kit for biomedical signals recording during ergometric test," in *17th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, 2020, pp. 1–6.

[120] H. P. D. Silva, J. Guerreiro, A. Lourenço, A. Fred, and R. Martins, "Bitalino: A novel hardware framework for physiological computing," in *PhyCS 2014 - Proceedings of the International Conference on Physiological Computing Systems*. SciTePress, 2014, pp. 246–253.

[121] K. Yelamarthi, M. S. Aman, and A. Abdelgawad, "An application-driven modular iot architecture," *Wireless Communications and Mobile Computing*, vol. 2017, 2017.

[122] K. Mikhaylov and J. Petäjäjärvi, "Design and implementation of the plug&play enabled flexible modular wireless sensor and actuator network platform: Design and implementation of plug&play enabled modular wsan platform," *Asian Journal of Control*, vol. 19, 2017.

[123] K. Mikhaylov and A. Paatelma, "Enabling modular plug&play wireless sensor and actuator network nodes: Software architecture," in *2015 IEEE SENSORS*, 2015, pp. 1–4.

[124] N. Tantitharanukul, K. Osathanunkul, K. Hantrakul, P. Pramokchon, and P. Khoenkaw, "MQTT-Topics Management System for sharing of Open Data," in *2nd Joint International Conference on Digital Arts, Media and Technology 2017: Digital Economy for Sustainable Growth, ICDAMT 2017*, 2017, pp. 62–65.

[125] United States Environmental Protection Agency, "Why save water? Statistics and Facts," [Online] Available: https://www.epa.gov/watersense/statistics-and-facts, (visited 21/03/2021). [Online]. Available: https://www.epa.gov/watersense/statistics-and-facts

[126] J. Alves Coelho, A. Glória, and P. Sebastião, "Precise Water Leak Detection Using Machine Learning and Real-Time Sensor Data," *IoT*, vol. 1, no. 2, pp. 474–493, 12 2020.

[127] Botn'Roll, "Water Flow Sensor YF-B2," [Online] Available: WaterFlowSensorYF-B2, (visited 23/10/2020). [Online]. Available: https://www.botnroll.com/en/biometrics/2543-water-flow-sensor-yf-b2.html

[128] A. Glória, J. Cardoso, and P. Sebastião, "Sustainable irrigation system for farming supported by machine learning and real-time sensor data," *Sensors*, vol. 21, no. 9, 5 2021.

[129] DFRobot, "Weather Station Datasheet," 2019, [Online] Available: https://wiki.dfrobot.com/Weather_Station_with_Anemometer_Wind_vane_Rain_bucket_SKU_SEN0186, (visited 21/02/2021).

[130] A. Glória, P. Sebastião, C. Dionísio, G. Simões, and J. Cardoso, "Water management for sustainable irrigation systems using internet-of-things," *Sensors (Switzerland)*, vol. 20, no. 5, 3 2020.

[131] J. Cardoso, A. Glória, and P. Sebastião, "A Methodology for Sustainable Farming Irrigation using WSN, NB-IoT and Machine Learning," in *5th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA20)*, 2020.

[132] M. Panda, "Intelligent Data Analysis for Sustainable Smart Grids using Hybrid Classification by Genetic Algorithm based Discretization," *Intelligent Decision Technologies*, vol. 11, 2017.

[133] S. Makonin, "AMPds2: The Almanac of Minutely Power dataset (Version 2)," 2016. [Online]. Available: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/FIE0S4

Appendices

APPENDIX A

Learning System Scenarios

This appendix presents a detailed description of the scenarios used in the Machine Learning classification and regression tests, in order to assess the best models and configurations to use in IoT projects. For each of them the scenario goal, dataset used and how it was gathered is described.

A.1. HVAC System Operation Detection

The first scenario was a classification scenario designed to identify if a HVAC system is turned ON inside an office space, based on a network of smart sensors collecting temperature and humidity data. This system intends to identify if the HVAC is turned ON and if it really needs to be, in order to improve energy efficiency and create a better environment for workers, while reducing costs.

The dataset used, as explained in [**83**], contains data from 6 different nodes inside a $42m^2$ office from two months of operation, corresponding to around 1000 samples collected, with the features presented in Table A1.

Table A1. HVAC Dataset

Feature	**Description**
Sensor_n	Value from Sensor Node (see Table A2)
Timestamp	Timestamp from collection
HVAC	HVAC state (ON/OFF)

Table A2. HVAC System Sensor Nodes

Sensor_n	**Location**
1	Window (First Floor)
2	Backoffice (First Floor)
3	Stair
1	Window (Second Floor)
1	Back Office (Second Floor)

The goal of this scenario is to classify whether the HVAC system is ON based on temperature and humidity data, as they are gathered in real time by the sensors.

A.2. Water Leak Detection

The second classification scenario under study was the use of WSN to predict leaks in water distribution pipes in agricultural fields, not only to detect when a leak occurred, but also the size and location.

The dataset used, created by [**84**], was composed using a small water distribution system in a laboratory, with three sensors placed on a pipeline, on which holes were carved, being the results classified based on location and size. The leaks were detected by a drop of water flow, captured by the sensor, when compared to the first or previous sensor in the system, allowing it to also detect the section, between two sensors, where the leak is encountered. The dataset also classifies the leak by size, according to Table A3.

TABLE A3. Leak Size Output [**84**]

Output	**Description**
0	No leaks in that section
1	Micro leaks in that section
2	Minor leaks in that section
3	Major leaks in that section

The dataset contains 5607 entries, with the features presented in Table A4.

TABLE A4. Leak Detection Dataset [**84**]

Feature	**Description**
id	ID of the data input
sensorID	Corresponding sensor
value	Value collected
average	Average from last 5 values for that sensor
diff_ref	Difference from Sensor 1
diff_sen	Difference from previous sensor
hasProblems	Leaks on that section

With this dataset a classification approach is intended to use water flow data, collected by the sensors in real time, to detect if a pipe has some sort of rupture.

A.3. Agricultural Irrigation Hour

Another classification problem studied was the use of ground sensor data and weather conditions, to predict the best time of day for water administration in agricultural fields, in order to reduce the consumption of water.

The dataset used, created by [**131, 85**], was gathered using a network of sensors that collected environmental and field data and complemented by values provided by the Instituto Português do Mar e da Atmosfera (IPMA) [**86**].

This data contains 105217 entries with a vast number of features, as can be seen in Table A5.

Table A5. Agricultural Irrigation Hour Dataset [**131, 85**]

Feature	**Description**
Year	Year of the observation
Month	Month of the observation
Day	Day of the observation
Hour	Hour of the observation
Temperature	Air Temperature registered [°C]
Relative_Humidity	Air Humidity registered [%]
Total_Precipitation_Low	Precipitation registered [mm/day]
Wind_Speed	Wind Speed registered [km/h]
Wind_Direction	Wind Direction registered [°]
Soil_Humidity	Soil Moisture registered [%]
Had_irrigation	Field irrigated [0/1]
Need_Irrigation	Field needs irrigation [0/1]
Is_Favorable	Conditions favorable for irrigation [0/1]
Suggested_Hour	Suggested irrigation hour [0–23]

The goal with this dataset is to use a classification technique to predict the next best hour of irrigation based on weather and field conditions gathered by the sensors in the field.

A.4. Weather Conditions

Another regression scenario tested was the prediction of weather conditions based on historical data. This is a common prediction example and can be very useful in our approach to more sustainable activities, mainly when linked with the previously presented agricultural scenarios, as weather is the main influence on those scenarios.

The dataset was provided by IPMA [**86**], and contains 52609 samples from hourly real time weather stations data from March 1st, 2014 to March 31st, 2019, for Lisbon, Portugal, containing the features presented in Table A6.

Table A6. Weather Conditions Dataset

Feature	**Description**
Latitude	Latitude of the field
Longitude	Longitude of the field
Altitude	Latitude of the field
Day	Day of the observation
Month	Month of the observation
Year	Year of the observation
Temperature_Max	Maximum temperature registered [°C]
Temperature_Min	Minimum temperature registered [°C]
Wind Speed	Average wind speed registered [km/h]
Precipitation	Total precipitation registered [mm/day]
Evapotranspiration	Evapotranspiration registered [mm/day]

The goal with this dataset is to use a regression technique to predict future temperature, humidity and precipitation using only a new timestamp.

A.5. SmartGrid House Consumption

The third regression scenario tested, accounts for the prediction of energy consumption inside a house based on smartgrid data. For that, multiple open-access datasets are available online, as M. Panda [**132**] analyzed. For our test, the Almanac of Minutely Power Data set (AMPDs) [**87**], that *"contains 1 minute aggregate meter readings as well as sub-metered readings from 19 individual circuits. Each reading includes measurements of voltage, current, frequency, power factor, real power, reactive power and apparent power. Furthermore, the aggregate gas and water consumption was also measured at 1 minute intervals, in addition to 1 individual usage for each utility."*, as M. Panda concluded [**132**]. From those 19 files the *"Electricity_WHE"* dataset [**133**] was chosen, containing the aggregate meter reading for the whole house.

The latest version of the dataset contains data for April 2012 to March 2014, with 1051200 records, with 12 features, from which only the Timestamp, that was converted from a unix timestamp to a date format (DD/MM/YY hh:mm) and then individualized into features (day, month, year, hour, minute); and the Instant Real Power, in Watt [W],

that was converted into a kilowatt-hour [kWh] value, were extracted. To complement the dataset, information about the energy contract was included, based on a tri-hourly contract with 3 periods of energy charge based on the day of the week, hour and season. The final dataset features are as displayed in Table A7.

Table A7. SmartGrid Consumption Dataset

Feature	**Description**
day	Timestamp day
month	Timestamp month
year	Timestamp year
hour	Timestamp hour
minutes	Timestamp minutes
weekday	Timestamp week day (Sunday - 1; Saturday - 7)
energyPeriod	Tri-hourly energy period
power	Real Power reading total meter reading [W]
energy	Energy consumption [kWh]

With this dataset, the intended goal is to be able to predict how much energy is being consumed in a certain timestamp based on smartgrid data. This can be incorporated in many smarthome solutions, to check if turning a new appliance will congest the network or even how much it will cost to the user and if turning it on another hour can potentially reduce the costs.

A.6. Communication Strength Signal

Finally, the last regression scenario falls under the device communication area, as this will be a key module of the developed solution. In this scenario the regression will be used to predict the quality of a signal, using different protocols, based on the location of the device.

For that, the NetBravo open-access dataset [**88**] was used, a crowd-sourced dataset composed by measurements of signal strength for Wi-Fi and 2G, 3G and 4G cellular based on coordinates location, all over Europe. The dataset contains 878289 entries, with the features presented in Table A8.

TABLE A8. Communication Strength Signal Dataset

Feature	**Description**
X	Coordinate X
Y	Coordinate Y
type	Communication Protocol
RSSI	Signal Strength

APPENDIX B

Machine Learning Algorithms Comparison Results

This appendix presents the results obtained in the comparison of Machine Learning Algorithms for data analysis, as described in Section 3.2.

B.1. Classification

TABLE B1. Default Classification Results

Algorithm	**Accuracy [%]**		
	HVAC	**Leak Detection**	**Irrigation Hour**
SVM	97.59	77.98	79.57
Random Forest	98.79	82.88	84.74
Neural Network	98.79	79.5	82.36
Decision Tree	97.59	81.28	76.83

TABLE B2. Hyper Classification Results

Algorithm	**Accuracy [%]**		
	HVAC	**Leak Detection**	**Irrigation Hour**
SVM	98.3	80.3	80.21
Random Forest	99.39	85.7	85.49
Neural Network	99.7	82.9	83.21
Decision Tree	98.79	83.5	79.65

TABLE B3. Cross-Validation Classification Results

Algorithm	**Accuracy [%]**		
	HVAC	**Leak Detection**	**Irrigation Hour**
SVM	98.19	78.62	79.3
Random Forest	98.5	83.06	84.6
Neural Network	99.34	78.96	80.2
Decision Tree	98.2	80.57	77.0

B.2. Regression

TABLE B4. Default Regression Results

Algorithm	**MAE**				
	Temperature [ºC]	**Humidity [%]**	**Precipitation [mm/day]**	**SmartGrid [kWh]**	**Comms [dBm]**
Linear Regression	6.123	11.961	0.169	0.172	8.957
Random Forest	0.851	4.813	0.071	0.038	7.314
Neural Network	6.168	10.515	0.11	0.151	8.938
Decision Tree	0.967	5.661	0.064	0.036	7.963

TABLE B5. Hyper Regression Results

Algorithm	**MAE**				
	Temperature [ºC]	**Humidity [%]**	**Precipitation [mm/day]**	**SmartGrid [kWh]**	**Comms [dBm]**
Linear Regression	6.123	11.961	0.169	0.172	8.957
Random Forest	0.786	4.825	0.072	0.035	7.187
Neural Network	6.159	9.462	0.169	0.142	8.027
Decision Tree	1.034	5.682	0.07	0.033	7.125

TABLE B6. Cross-Validation Regression Results

Algorithm	**MAE**				
	Temperature [ºC]	**Humidity [%]**	**Precipitation [mm/day]**	**SmartGrid [kWh]**	**Comms [dBm]**
Linear Regression	6.155	12.101	0.173	0.173	8.951
Random Forest	1.042	5.626	0.096	0.049	7.576
Neural Network	3.381	10.314	0.173	0.147	8.932
Decision Tree	1.269	6.519	0.095	0.045	8.35

APPENDIX C

Smart Node Schematics

This appendix presents a detailed description of the developed modules for the smart IoT node. For each of the modules the schematics, the Bill of Materials (BOM), the PCB layout and the final assembled board are presented.

C.1. Main Module

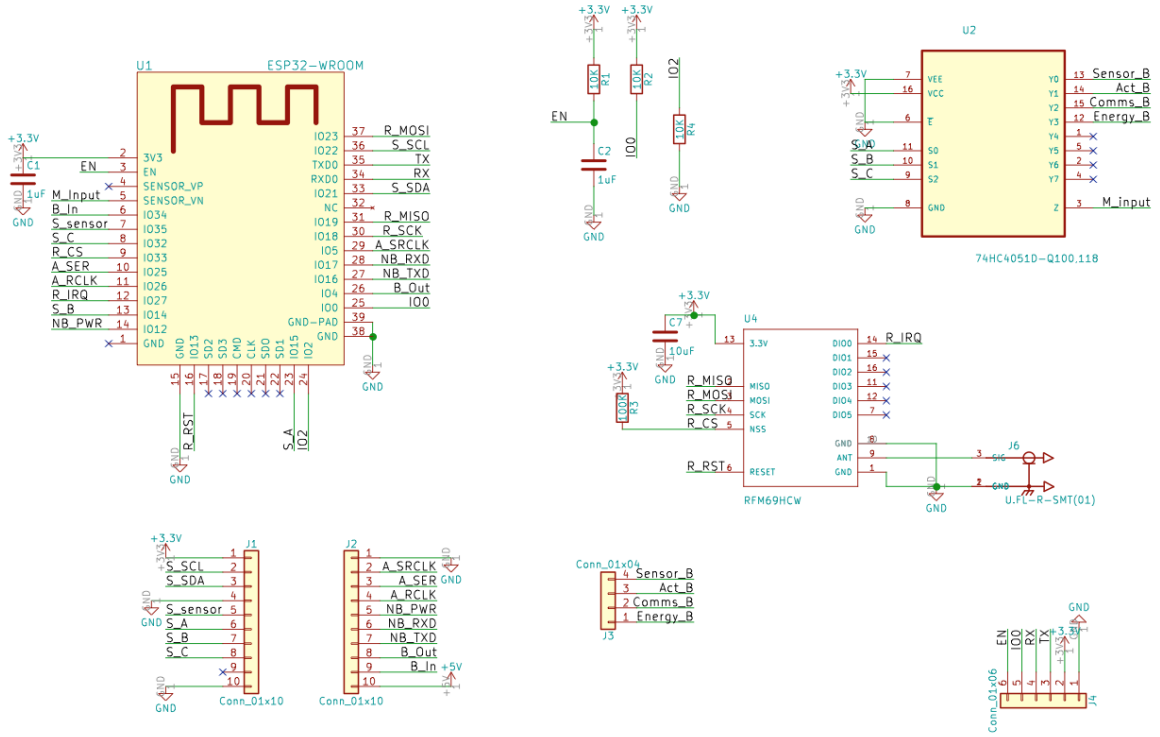
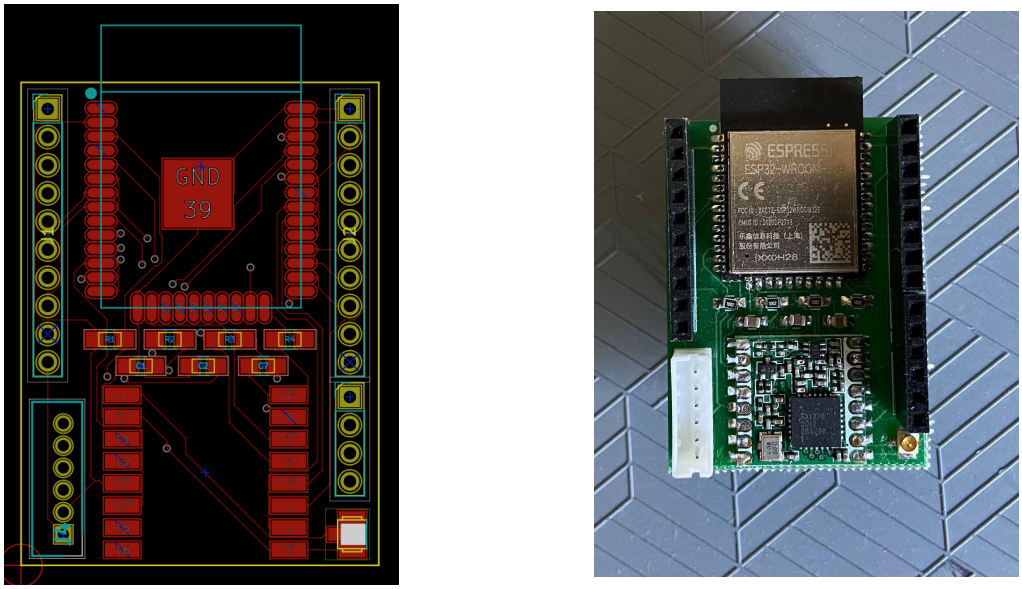


FIGURE C1. Main Module Schematics

TABLE C1. Main Module BOM

Label	Description	Label	Description
C1-C2	1μF Capacitor (SMD 0805)	R1-R2	10kΩ Resistor (SMD 0805)
C3	10μF Capacitor (SMD 0805)	R3	100kΩ Resistor (SMD 0805)
J1-J2	1x10 Male Stackable Header (2.54mm)	R4	10kΩ Resistor (SMD 0805)
J3	1x4 Male Stackable Header (2.54mm)	U1	ESP32-WROOM-32E
J4	1x6 Male JST PH Connector (2.00mm)	U2	74HC4051 (SOIC127)
J6	U.FL Male Connector	U3	RFM95W



(a) PCB Layout

(b) Assembled Node

FIGURE C2. Main Module Implementation

C.2. Sensor Module

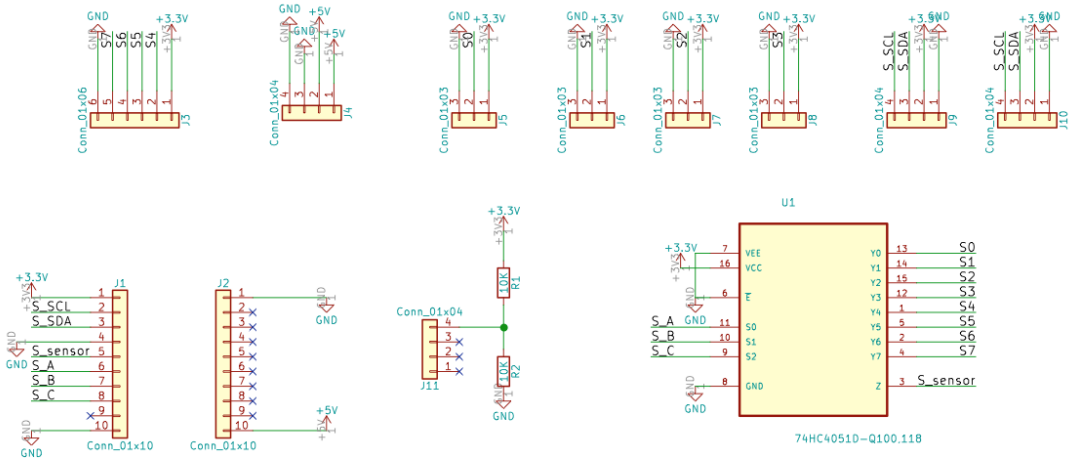
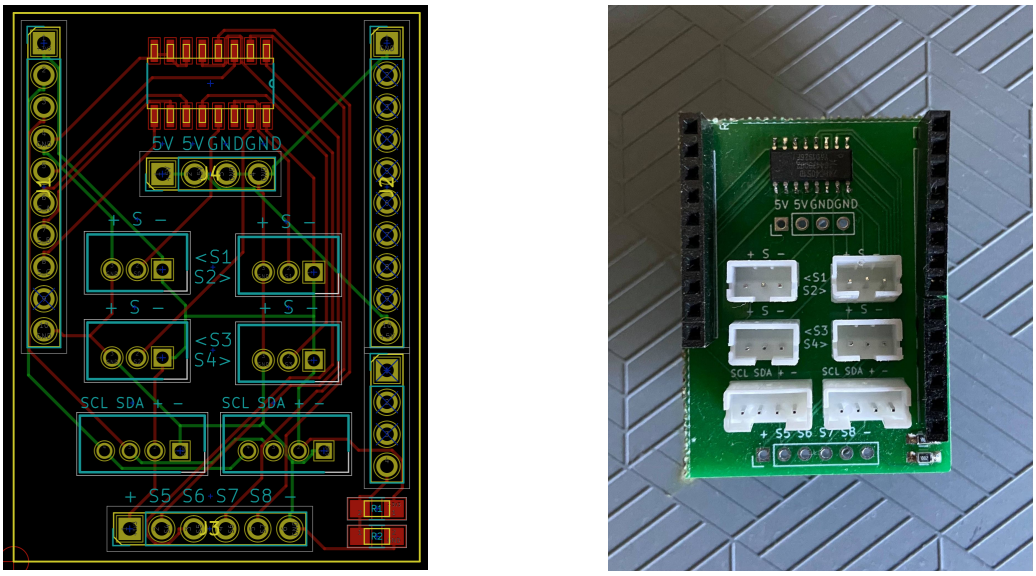


FIGURE C3. Sensor Module Schematics

TABLE C2. Sensor Module BOM

Label	Description	Label	Description
J1-J2	1x10 Male Stackable Header (2.54mm)	J9-J10	1x4 Male JST PH Connector (2.00mm)
J3	1x6 Male Header (2.54mm)	J11	1x4 Male Stackable Header (2.54mm)
J4	1x4 Male Header (2.54mm)	R1-R2	10kΩ Resistor (SMD 0805)
J5-J8	1x3 Male JST PH Connector (2.00mm)	U1	74HC4051 (SOIC127)



(a) PCB Layout

(b) Assembled Node

FIGURE C4. Sensor Module Implementation

C.3. Actuator Module

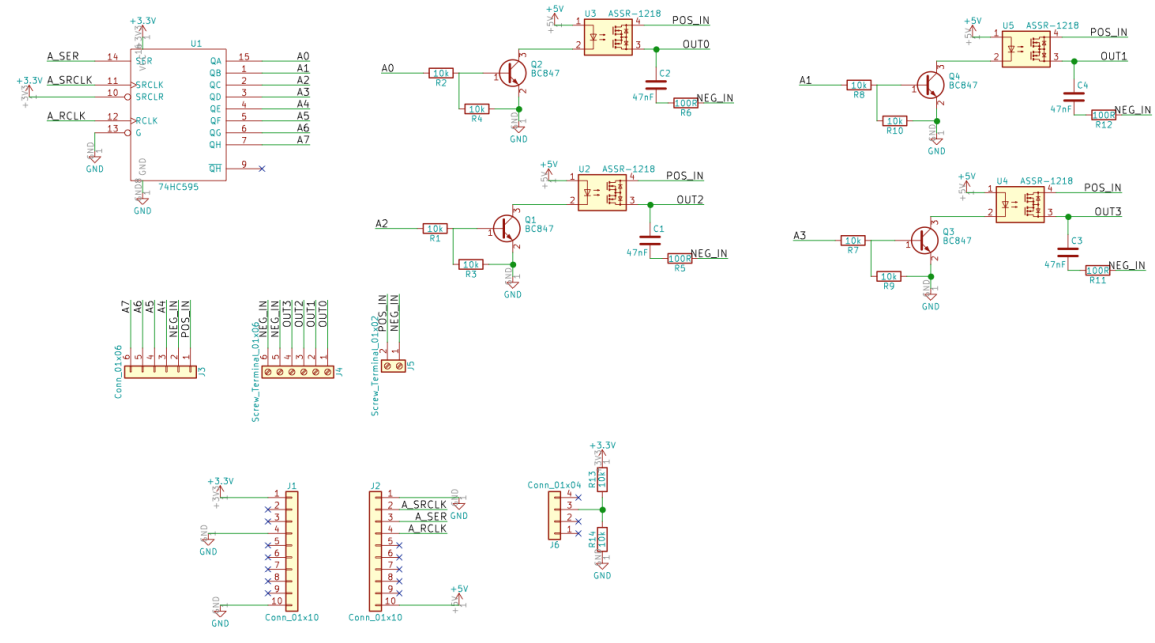
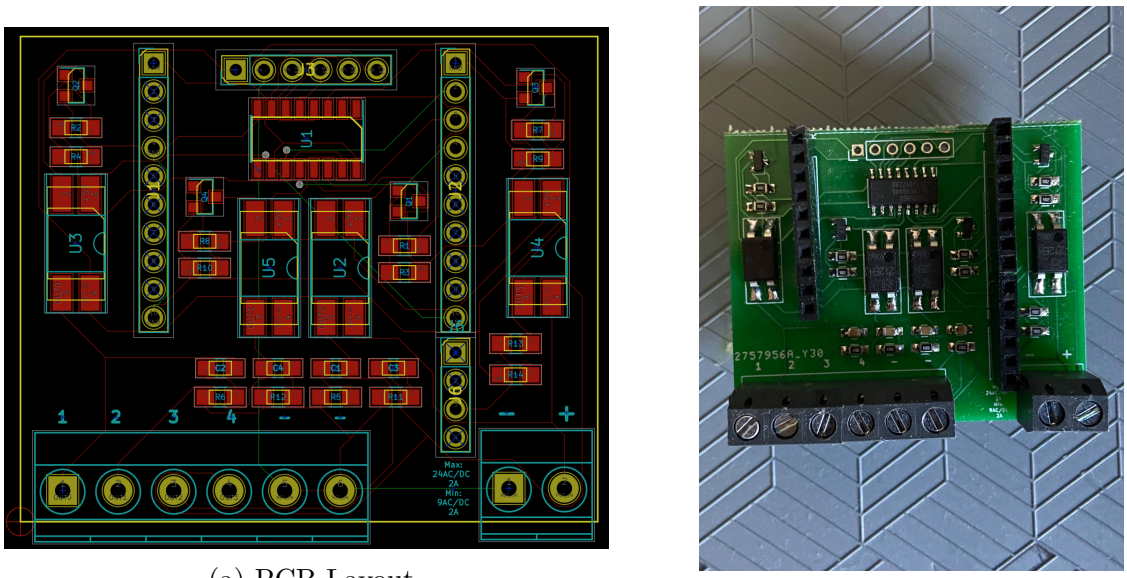


FIGURE C5. Actuator Module Schematics

TABLE C3. Actuator Module BOM

Label	Description	Label	Description
C1-C4	47nF Capacitor (SMD 0805)	R5-R6	100Ω Resistor (SMD 0805)
J1-J2	1x10 Male Stackable Header (2.54mm)	R7-R10	10kΩ Resistor (SMD 0805)
J3	1x6 Male Header (2.54mm)	R11-R12	100Ω Resistor (SMD 0805)
J4-J5	1x6 Terminal Block (5.00mm)	R13-R14	10kΩ Resistor (SMD 0805)
J6	1x4 Male Stackable Header (2.54mm)	U1	74HC595 (SOIC-16)
Q1-Q4	BC847 (SMD SOT-23)	U2-U5	AQY212EHAT (SSR DIP-4)
R1-R4	10kΩ Resistor (SMD 0805)		



(a) PCB Layout

(b) Assembled Module

FIGURE C6. Actuator Module Implementation

C.4. Battery Module

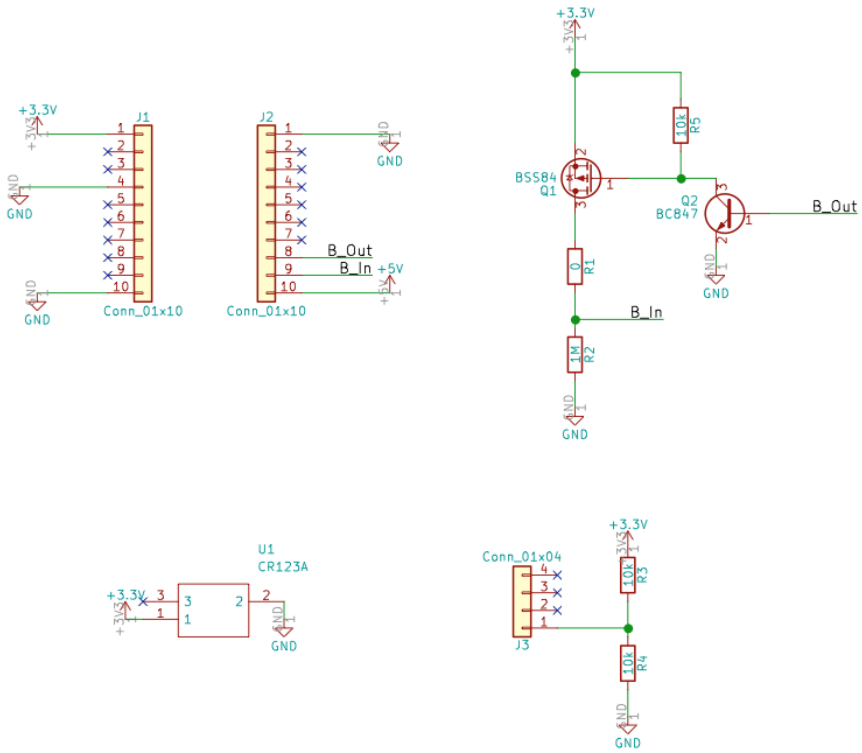
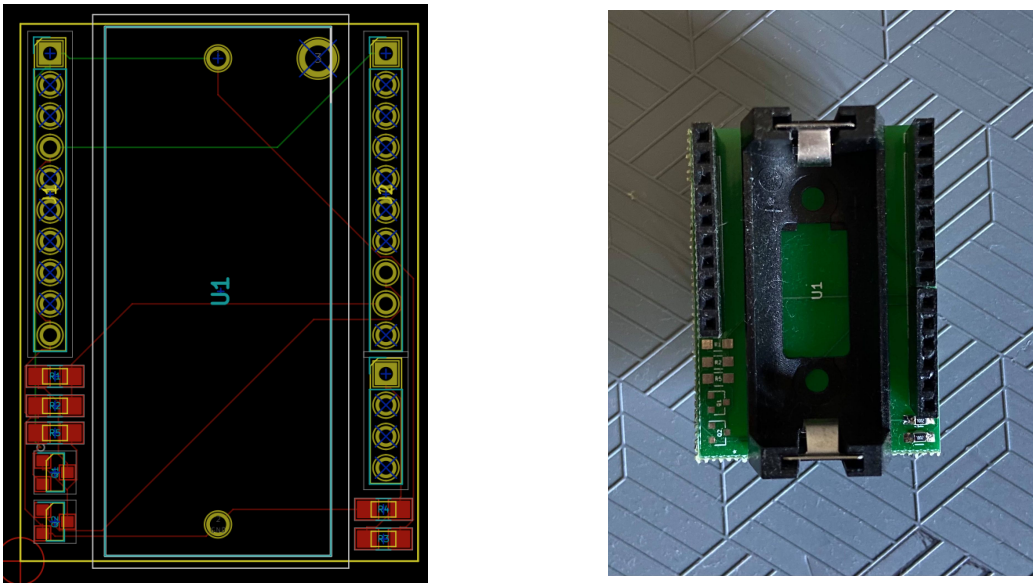


FIGURE C7. Battery Module Schematics

TABLE C4. Battery Module BOM

Label	Description	Label	Description
J1-J2	1x10 Male Stackable Header (2.54mm)	R1	0Ω Resistor (SMD 0805)
J3	1x6 Male Header (2.54mm)	R2	1MΩ Resistor (SMD 0805)
Q1	BSS84 (SOT-23)	R3-R5	10kΩ Resistor (SMD 0805)
Q2	BC847 (SOT-23)	U1	CR123A Battery Socket



(a) PCB Layout (b) Assembled Module

FIGURE C8. Battery Module Implementation

C.5. USB Module

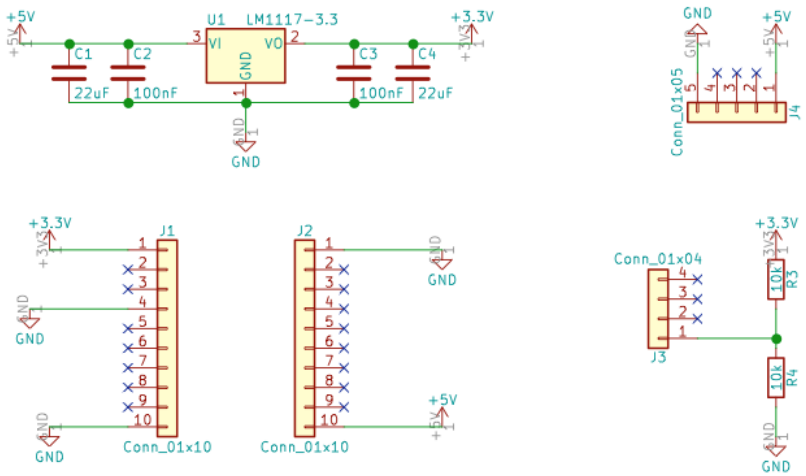
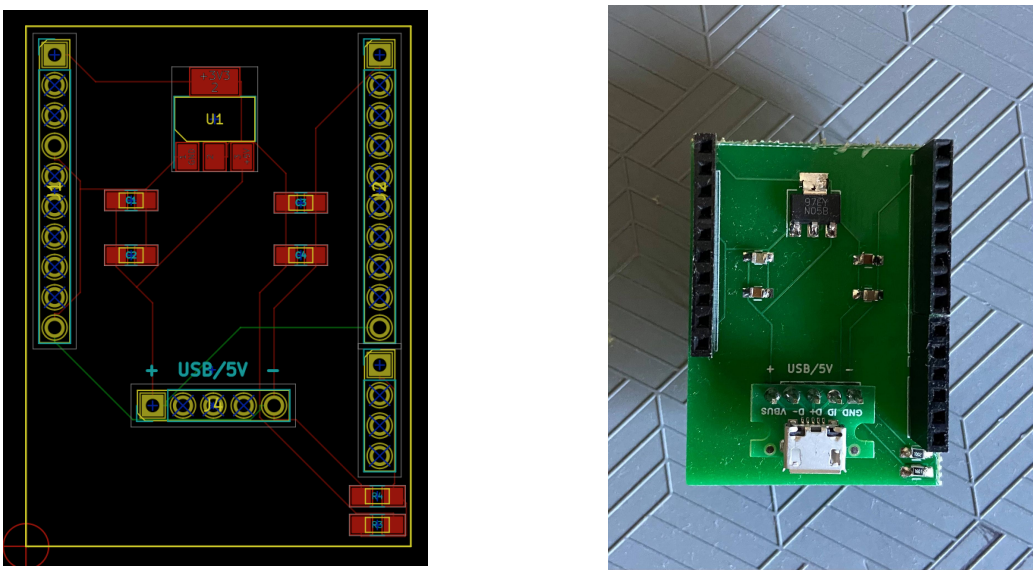


FIGURE C9. USB Module Schematics

TABLE C5. USB Module BOM

Label	Description	Label	Description
C1	22μF Capacitor (SMD 0805)	J3	1x4 Male Stackable Header (2.54mm)
C2-C3	100nF Capacitor (SMD 0805)	J4	USB Connector
C4	22μF Capacitor (SMD 0805)	R3-R4	10kΩ Resistor (SMD 0805)
J1-J2	1x10 Male Stackable Header (2.54mm)	U1	LM1117-3.3 (SMD SOT-223)



(a) PCB Layout

(b) Assembled Module

FIGURE C10. USB Module Implementation

C.6. AC/DC Module

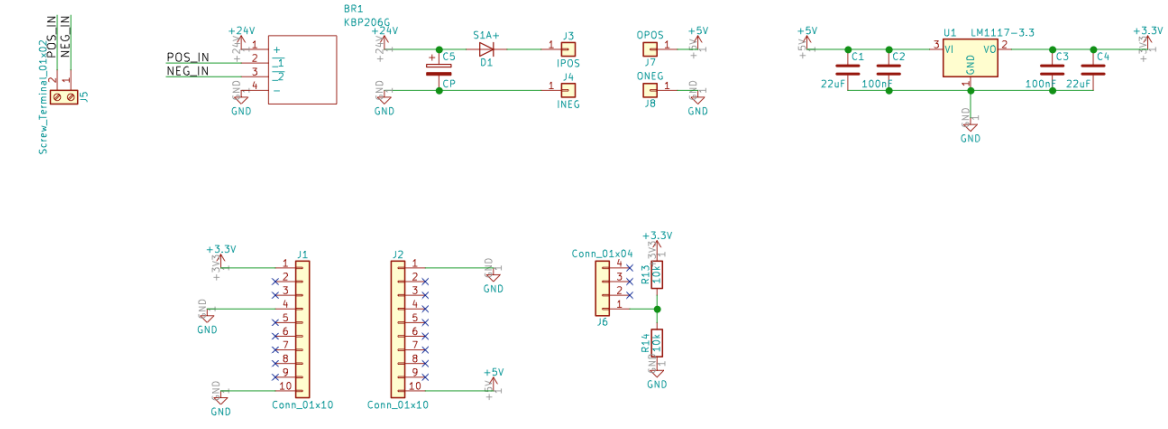
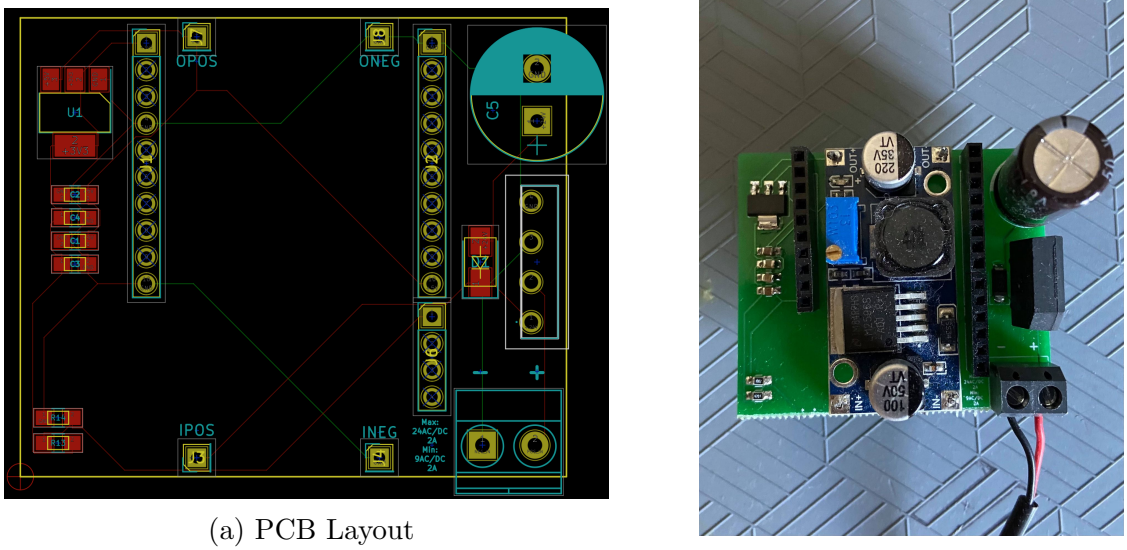


FIGURE C11. AC/DC Module Schematics

TABLE C6. AC/DC Module BOM

Label	Description	Label	Description
BR1	KBP206G	J1-J2	1x10 Male Stackable Header (2.54mm)
C1	22μF Capacitor (SMD 0805)	J3/4/7/8	LM2596 Step Down Converter
C2-C3	100nF Capacitor (SMD 0805)	J5	1x2 Terminal Block (5.00mm)
C4	22μF Capacitor (SMD 0805)	J6	1x4 Male Stackable Header (2.54mm)
C5	1000μF Radial Capacitor (THT 5.0mm)	R13-R14	10kΩ Resistor (SMD 0805)
D1	S1A+ Recovery Diode (SMA)	U1	LM1117-3.3 (SMD SOT-223)



(a) PCB Layout

(b) Assembled Module

FIGURE C12. AC/DC Module Implementation