UNIVERSITY OF PADOVA

Department of Management and Engineering
Ph.D. Course in Mechatronics and Product Innovation Engineering

Ph.D. THESIS

Industrial Internet of Things for Smart and Innovative Measurement Systems.

Supervisor: Prof. Federico Tramarin
Co-Supervisor: Prof. Stefano Vitturi

Candidate: Tommaso Fedullo

ACADEMIC YEAR: 2022/2023

"Our life is a constant journey...

...The landscape changes, the people change, our needs change, but the train keeps moving. Life is the train, not the station."[1]

Dedicated to the man who taught me to always catch trains on time.

[1] Paulo Coelho

Abstract

In the last years, the development of novel techniques within the Internet of Things (IoT) paradigm is reshaping the way measurement systems are designed and developed. Several applications require to derive measurements in harsh environments, where nodes are moving or unreachable. A common example is represented by the novel smart factory where, for example, moving robots, additive manufacturing processes, critical control processes and safety-related activities need for accurate and real-time measurement activities. These novel smart measurement systems allows to take continuous and thorough measurements, simultaneously on wide areas. This brings to the development of smart and integrated sensors networks that should meet the requirements coming from the industrial environment and, generally speaking, the Instrumentation and Measurement (I&M) field, like timeliness, determinism, low-latency and robust behavior. In this context, one critical aspect is surely related with communication, especially the wireless one. In this Ph.D. thesis, several wireless technologies like Wi-Fi, Long Range (LoRa) and 5G are compared. From the comparative analysis between different wireless technologies, it is possible to conclude that each of them has strengths and weaknesses. For this reason, the choice of a *one fits all* communication technology is challenging, and a possible solution is represented by the Time Sensitive Networking (TSN) set of protocols. TSN aims to develop a set of algorithms that comprehensively are able to provide high levels of determinism, and several studies addresses the usage of TSN on top of Wi-Fi. Despite this, the communication technology must be chosen by carefully analyzing the specific application. For example, LoRa networks are the best candidates when long communication range and low en-

ergy consumption is needed. For example, in the following, an additive manufacturing test case, where sensors are embedded into artifacts, will be analyzed. In this scenario, energy consumption becomes the most important performance indicator. Wi-Fi or 5G can, on the other hand, handle Ultra Low Latency (ULL) communication. Indeed, talking about communication delay (or, equivalently of data rate) and accordingly to the simulations presented in the following, 5G can send data within 10 - 15 ms while LoRa data rates are very low due to inter message delays of even some seconds.

The aforementioned wireless communication strategies, must be carefully analyzed and optimized, foreseeing to meet the stringent requirements coming from the industrial field. For this reason, in this thesis, novel optimization strategies are proposed for Wi-Fi and LoRa, aiming at adapting them for real-time smart measurement systems. Particular attention is given to Machine Learning (ML) strategies, as they represent key techniques in the IoT context. Results demonstrated the effectiveness of the proposed optimization techniques, and the applicability of these protocols for IoT-based smart measurement systems. Moreover, optimized communication strategies must be used in conjunction with efficient and accurate post-processing techniques. Indeed, post processing activities are of fundamental importance to derive measurements from data and to assure a certain degree of data integrity. For this reason, some ML-based post processing techniques have been developed and analyzed as a secondary research project. As experienced in this work, ML can improve the performances of these novel smart measurement systems, but at present a big challenge is to characterize the ML techniques from a metrological point of view, for example by analyzing the measurement accuracy. As a matter of fact, by both analyzing the current literature in the field and the results obtained from several experiments, the aim of this study is twofold. First of all, novel techniques to handle IoT-based measurement systems are presented. Secondly, this work aims at underlining both the current progressions and the main research challenges in the field.

"A journey well shared is a journey well enjoyed." [2]

Acknowledgements

First of all, I have to make a special thank to my family, in particular to my sister, Laura, with who I fought many battles. Actually, I do not really know if we came out winners or losers. We came out, somehow.

In these three years I met a lot of people, and I want to make a special thank to all of them. First of all, to my supervisor, Prof. Federico Tramarin from who I learned how to be a good researcher. A big thank to all friends, collegues and professors I met, with who I shared a lot of professional and life moments. I want to start with Prof. Stefano Vitturi and Prof. Luigi Rovati. Then, a big thank to the optoboys, my collegues in Modena, especially to Ettore, Giovanni and Valentina. Last but not least, thanks to my Swedish Gang: Sujash, Farnaz and Roberto.

[2] Unknown Author

Contents

List of Figures

List of Tables

Glossary

ACA Anterior Chamber Angle. 117, 118, 120, 123

ACK acknowledgement. 55

ADR Adaptive Data Rate. 72, 75, 95, 98, 100, 105

AI Artificial Intelligence. 3, 5, 10, 12, 13, 21, 22, 23, 24, 117, 133

AME Absolute Mean Error. 28

AP Access Point. 55, 80

ARF Automatic Rate Fallback. 56, 57, 59, 78

AVB Audio Video Bridging. 65

BW BandWidth. 72, 74, 109

CAD Channel Activity Detection. 106, 110

CIM Computer Integrated Manufacturing. 2, 3

CNN Convolutional Neural Networks. 11, 12, 34, 120, 121, 122, 125, 127

CPS Cyber Physical System. 1, 15, 53

CR Coding Rate. 72, 109

CSMA Carrier Sense Multiple Access. 106, 107, 110, 113, 114

CSMA/CD Carrier Sense Multiple Access with Collision Detection. 55

CSS Chirp Spread Spectrum. 72

CTS Clear To Send. 107, 108

CW Contention Window. 106

D2D Device-to-Device. 107, 110, 113, 114, 143

DC Duty Cycle. 5, 72, 73, 74, 95, 105, 106, 107, 111, 114, 134

DCF Distributed Coordination Function. 55

DER Data Extraction Rate. 15, 96, 97, 98, 100, 105, 143

DIFS Distributed Inter-Frame Space. 106

DL Deep Learning. 12, 33

DLL Data Link Layer. 54, 65, 69

DLP Digital Light Projector. 118, 119

E2E End-to-End. 110, 113, 114

ECDF Experimental Cumulative Distribution Function. 82, 85, 87

ED End Device. 71, 72, 75

FARF Fast retransmission rate ARF. 57, 59, 81, 82, 84

FN False Negatives. 43

FP False Positives. 43, 127

FRER Frame Replication and Elimination for Reliability. 69

GRRAA Green RRAA. 97, 98, 100, 105

GUI Graphical User Interface. 119

GUM Guide to the Expression of Uncertainty in Measurement. 14, 133

GW GateWay. 71, 72, 73, 96, 98, 100, 105, 106, 107, 109, 110, 114

hybAC hybrid ALOHA-CSMA. 106, 110, 112, 113

hybACR hybrid ALOHA-CSMA strategy with Relaying capabilities. 107, 110, 113, 114

I&M Instrumentation and Measurement. 3, 5, 7, 10, 17, 21, 27, 51, 79, 95, 139

IIoT *Industrial Internet of Things*. 1, 12, 13, 15, 17

IoM Internet of Measurements. 10, 11, 12, 15, 16, 17, 53, 70, 72, 77, 93, 95, 105, 117, 130, 134, 135, 137, 138

IoT *Internet of Things*. 1, 3, 4, 5, 7, 8, 9, 10, 12, 14, 15, 17, 18, 23, 70, 77, 96, 117, 133, 135

IoV Internet of Vehicles. 10

IoWT Internet of Wearable Things. 9

ISM Industrial Scientifical Medical. 72, 73

ISO International Organization for Standardization. 14

LAN Local Area Network. 3, 66, 69, 77

LES LoRaEnergySim. 98, 110

LoRa Long Range. 4, 5, 51, 54, 70, 71, 72, 75, 93, 95, 105, 106, 107, 111, 134

LoRaWAN LoRa Wide Area Network. 70, 71, 72, 75, 106

LPWANs Low Power Wide Area Networks. 70

LR Learning Rate. 36, 44, 45

MAC Medium Access Control. 64, 66, 78, 79, 84, 85, 89, 105

MCS Modulation Coding Scheme. 81

MCT Minimum Cycle Time. 89, 90

MDP Markov Decision Process. 40, 96

ME Mean Error. 28

ML Machine Learning. 3, 4, 5, 10, 12, 17, 24, 25, 27, 31, 36, 39, 42, 43, 117, 118, 119, 120, 121, 122, 130, 133, 134

MLPNN Multi Layered Perceptron Neural Networks. 32

MMH Maximum Marginal Hyperplane. 36

MRS Multi Rate Support. 56

NN Neural Network. 25, 27, 31, 32, 33, 36, 43, 44, 45, 121, 130

NS Network Server. 71, 72, 75

OSI Open System Interconnection. 141

PACG Primary Angle Closure Glaucoma. 117, 118

PC Personal Computer. 78, 79

PCA Principal Component Analysis. 27

PCF Point Coordination Function. 55

PER Packet Error Rate. 60, 79

PLC Programmable Logic Controller. 79, 93

PMU Phasor Measurement Units. 138, 140

PTB Physikalisch-Technische Bundesanstalt. 14

QoS Quality of Service. 16, 66, 68, 69

RA Rate Adaptation. 54, 77, 82, 84, 87, 95, 133, 134

RAA Rate Adapation Algorithm. 56, 77, 78, 79, 80, 81, 85, 87, 134

RAN Radio Access Network. 141

RF Random Forest. 25, 27, 36

RFId Radio Frequency Identification. 7

RL Reinforcement Learning. 5, 24, 27, 39, 40, 41, 95, 96, 97, 134

RMSE Root Mean Squared Error. 28

RRAA RL-based Rate Adaptation Algorithm. 96, 98, 100, 105

RSIN Rate Selection for Industrial Networks. 60, 63, 64, 79, 81, 82, 84, 87, 89, 90

RSIN-L RSIN Light. 79, 82, 83, 84, 89, 90

RSIN-E RSIN with Estimation. 64, 82, 83, 84, 89

RTE Real Time Ethernet. 14, 52, 53, 65

RTS Request To Send. 107, 108

RTT Round Trip Time. 87

SARF Static retransmission rate ARF. 57, 59, 82, 84, 87

SCC Sparse Categorical Crossentropy. 126

SF Spreading Factor. 72, 74, 75, 96, 97, 100, 106, 107, 109, 110, 111, 112, 114

SGD Stochastic Gradient Descent. 125

SNR Signal to Noise Ratio. 60, 64, 79, 96, 97, 98, 107

SR Symbol Rate. 72, 73, 74

SSE Sum of euclidean Squared Errors. 37

SVM Support Vector Machine. 25, 27, 36, 121

TDMA Time Division Multiple Access. 54

TG Task Group. 65

TP True Positives. 43, 75, 127

TP Transmission Power. 72, 75, 96, 97, 100

TSN Time Sensitive Networking. 3, 5, 15, 64, 65, 66, 68, 69, 70, 133, 141

UD User Device. 141, 142

UDP User Datagram Protocol. 68, 69

ULL Ultra Low Latency. 70, 142

UPF User Plane Functions. 142

VANETs VEhicle ad-hoc NETworks. 10

VBMs Vision Based Measurement systems. 3, 11, 122, 130

VH Van Herick. 118, 119, 120, 121, 131

WLAN Wireless Local Area Network. 54

WNCS Wireless Networked Control Systems. 53

WNIC Wireless Network Interface Cards. 78, 81, 87

Introduction

In recent years the plan Industry 4.0 is fostering a significant transformation within the field of industrial automation and control. In this context, the industrial and process automation fields can also be seen as a significant instance of the Cyber Physical System (CPS) concept. The CPS, collectively refers to those systems where a number of computational elements is in charge of connecting the virtual to the physical world, realizing computation, control and communication tasks [1]. These paradigms point to the realization of the so-called *Industrial Internet of Things* (IIoT). The latter foresees to transfer the *Internet of Things* (IoT) also in the industrial context, enabling the interconnection of anything, anywhere and at any time. In particular, it comprehensively groups several technologies and protocols (such as wireless communication, post processing techniques, smart devices), aiming at the smart interconnection of sensors, actuators and machines in the field level of the novel smart industry.

At present, the need for ever smarter industrial plants inevitably foresees the development of IoT-based smart measurement systems. Indeed, the need for continuous and thorough measurements, to be taken simultaneously on wide areas and in real-time, inevitably requires the usage of IoT systems. However, the scenarios where sensors and instruments are placed on a wide area or on mobile systems remain extremely challenging. Especially, the wireless (or possibly the hybrid wired-wireless) networks should be made capable of handling safety and time-critical traffic. More precisely, requirements of the industrial domain pose additional constraints in terms of reliability and real-time performance, just to mention a few, which require solutions yet to come. Moreover, the industrial measurement systems devoted to acquire and analyze

information from the field, are not anymore stand-alone systems, but they need to be integrated within the whole smart factory. Even before the advent of the well-known industry 4.0 concept, the need for integrated communication systems had been already underlined, for example by the Computer Integrated Manufacturing (CIM) concept. The latter has been somehow superseded by industry 4.0 as there is a gap between CIM requirements and industry 4.0 ones [2]. Indeed, authors of [2] identify several characteristics that are present in the industry 4.0 concept, but not in the CIM, such as, between others, predictive maintenance, self configuration and flexibility. Despite this, CIM still represents a good model to both represent the industrial levels, and to underline the needs from the communication perspective. Indeed, the latter vary depending on the specific level of the smart factoy. Moreover, the information coming from the field must be integrated with the data coming from the upper levels, that support the production (e.g. marketing, planning etc). The factory levels, according to CIM are summarized int he so-called automation pyramid, depicted in Figure 1.

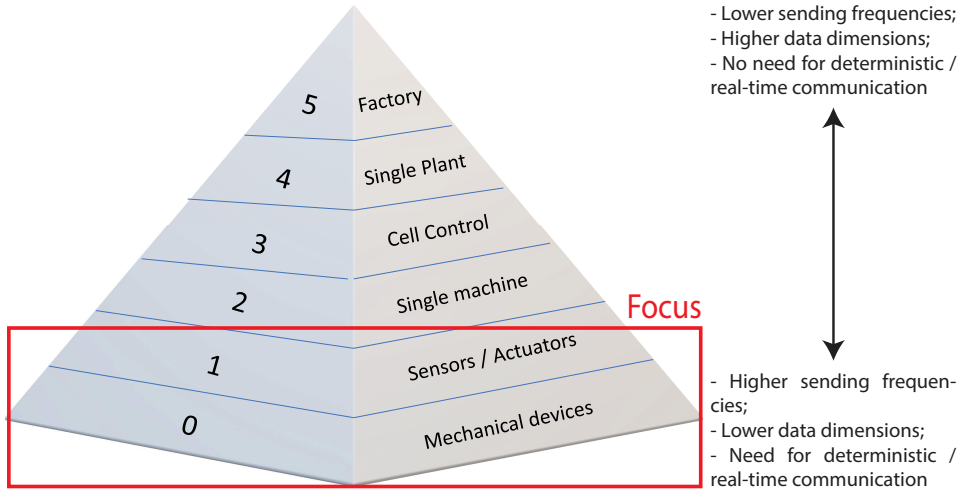

Figure 1: The automation pyramid.

As can be seen from the figure above, different factory levels introduce different

communication requirements. The focus of the thesis will be on the two lowest level of the automation pyramid, in particular in optimizing the wireless communication networks to cope with the depicted requirements. The requirements coming from the industrial field (lower levels of the CIM pyramid), including timeliness and real-time behavior, can also be generalized and extended to other application fields, where real-time and smart measurement systems are needed (e.g. energy systems, aeronautical applications, etc). As a matter of fact, there is the need for sensors networks capable to accurately derive measurements in a real-time manner. For this reason, the Instrumentation and Measurement (I&M) field needs to introduce, implement and analyze a plethora of novel technologies.

In this context, the I&M field needs to embody some of the enabling technologies of the IoT systems, like for example Artificial Intelligence (AI), Machine Learning (ML) and high performing communication networks. This topic is still challenging, as the aforementioned techniques have been developed for general-purpose systems, where timeliness and measurement accuracy are not key requirements. For example, the latest generation of wireless technologies, as the LPWANs, WiFi6, 5G etc, represent an incredible opportunity. Despite this, their optimization (for instance in terms of timeliness, determinism, energy consumption, and data extraction rate) and their analysis from a metrological perspective remain an important goal. In this context, the novel Time Sensitive Networking (TSN) set of standards becomes very attractive. As we will analyze in the following, TSN aims to provide determinism, by defining a set of protocols built on top of the Ethernet standard. For this reason, TSN can be in principle built on top of also different Local Area Network (LAN)s, and TSN over Wi-Fi has already been proposed and analyzed. Furthermore, to obtain more accurate measurements and improve data exchange capabilities, novel techniques have been considered recently. Between others, ML approaches, are gaining a greater importance. For instance, Vision Based Measurement systems (VBMs)s are nowadays applied to several different fields, where measures must be derived from images. Another emerging field that could be relevant for sensor applications is the use of ML techniques for improving compressive sensing. In light of these considerations, it is also clear the need for accurate metrics to objectively evaluate both the obtainable communication performance

(in terms for example of correctly received packets, determinism or latency) and the information quality. Indeed, also the post processing, or the analysis, of the acquired data have to be precisely taken into account. The combination of such new technologies can have a very positive impact on instrumentation and sensor networks, especially in the era of the so called *digital twins*. In this context, the usage and the development of simulation frameworks has to be complemented by setting up meaningful experimental testbed to validate the results (and the compliance with standards and regulations).

Authors of [3] precisely summarized the challenges and enabling technologies for the development of these innovative smart measurement systems. In particular, it is possible to identify several advantages on the usage of IoT based measurement systems:

1. Possibility to perform a continuous and thorough measurement activity;
2. Possibility to take simultaneous measurements on wide areas;
3. Possibility to perform a timeliness and real-time measurement activity;
4. Possibility to smartly and efficiently process the acquired measurements, ensuring also the data integrity;
5. Possibility to take measurements on challenging environments, even connecting unreachable or moving nodes.

In this context, it is possible to summarize the research challenges on the topic as it follows:

1. Development of real-time and low-latency communication networks;
2. Real-Time processing of the acquired data;
3. Metrological characterization of IoT and ML approaches to measurements.

In this work, the development of real-time and low-latency communication networks is deeply addressed. In particular, the optimization of Wi-Fi and Long Range (LoRa) networks is extensively studied, by means of both experimental testbeds and

realistic simulations. In particular, parameter adaptation strategies for both Wi-Fi and LoRa have been analyzed, and new Reinforcement Learning (RL) - based ones are proposed. Furthermore, the Duty Cycle (DC) limitation on LoRa networks has been addressed, and novel strategies to avoid it are proposed. As a matter of fact, it will be possible to demonstrate that the proposed techniques allow the usage of these general-purpose networks also in the I&M field. It is worth observing that the aforementioned wireless protocols have very different characteristics, so that different I&M applications may use different communication strategies. For example, if higher sending frequencies are needed, and low-energy consumption is not an issue, Wi-Fi outperfoms LoRa and vice-versa. Moreover, some simulations on 5G networks and an analysis of TSN (also over Wi-Fi) are added, in order to be able to compare the most appealing communication strategies on the field.

As a secondary project, the development of some ML - based post processing techniques have been addressed. In particular, I developed some ML-based techniques to classify images coming from a smart measurement system for ophthalmic applications. Several ML techniques have been analyzed and compared, for example: K-means, space vector machines and neural networks. The best ones in terms of classification performances have been finally selected. In this context, we will see that one of the main challenges is to study the accuracy of ML techniques, being this a big limitation on their usage for safety applications and in general in the I&M scenario. Indeed, ML techniques are usually considered, from a user perspective, a black-box approach. In facts, they are used when a system model is not available. This approach leds to a non predictable behavior, that needs further investigation when applied to safety related applications or in general when a certain measurement accuracy is needed.

In the following, the first chapters will introduce the most important technologies addressed in this work. In particular, Chapter 1 will introduce both the IoT concept and the requirements for smart measurement systems. Chapters 2 and 3 respectively introduce the two considered enabling technologies in the context of IoT - based measurement systems: AI and communication networks. In these chapters, a general overview of the topics is given, the state of the art presented, with a special focus on the concepts needed more for the experimental part. Afterwards, experimental

results are presented, and the conclusions are given.

Chapter 1

The IoT in the I&M Field

"When we talk about the Internet of Things, it's not just putting RFId tags on some dumb thing so we smart people know where that dumb thing is. It's about embedding intelligence so things become smarter and do more than they were proposed to do.[1]*"*

Nowadays, the IoT is literally spreading in every human activity, becoming popular also for the non insiders. The first formulation of the IoT concept comes from a presentation by Kevin Ashton in 1999 [4], and was strictly related to Radio Frequency Identification (RFId) devices. The IoT formulation evolved during the years, and now comprise all the systems where objects are interconnected and acquire computational capability, becoming someway intelligent and cooperative. At present, everyday life greatly benefits from the IoT concept, allowing people to smartly control the house, the car and also the body health, only to name a few. How comfortable is it to directly "ask" to your house to make the coffee in the morning or to control the temperature and the lights of your new smart house?

The aim of this chapter is to analyze the enabling technologies of the IoT, together with its main advantages and applications. Moreover, a contextualization on the I&M field is given.

[1] Nicholas Negroponte, Computer Scientist

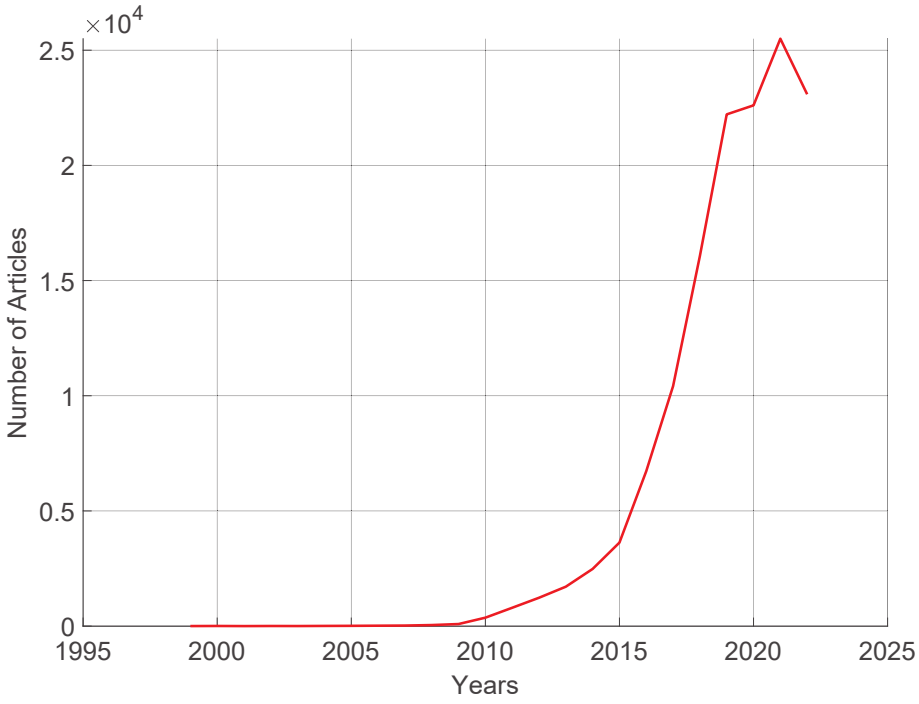

Figure 1.1: Scopus research on IoT: number of published articles per year.

1.1 Main Fields of Application and architecture

The growing interest for IoT systems is already visible to everyone, as at present almost everyone has experienced the benefits coming from this innovative paradigm. In parallel, also the research and academic interest on this technology grown a lot, as demonstrated by Figure 1.1.

As it is possible to see, the number of published articles on IoT during the years is grown exponentially. A deeper analysis on the published article can reveal also the main fields of application. Figure 1.2 represents the number of articles published on IoT per subject area.

As expected, most of the articles found on Scopus about IoT are categorized in

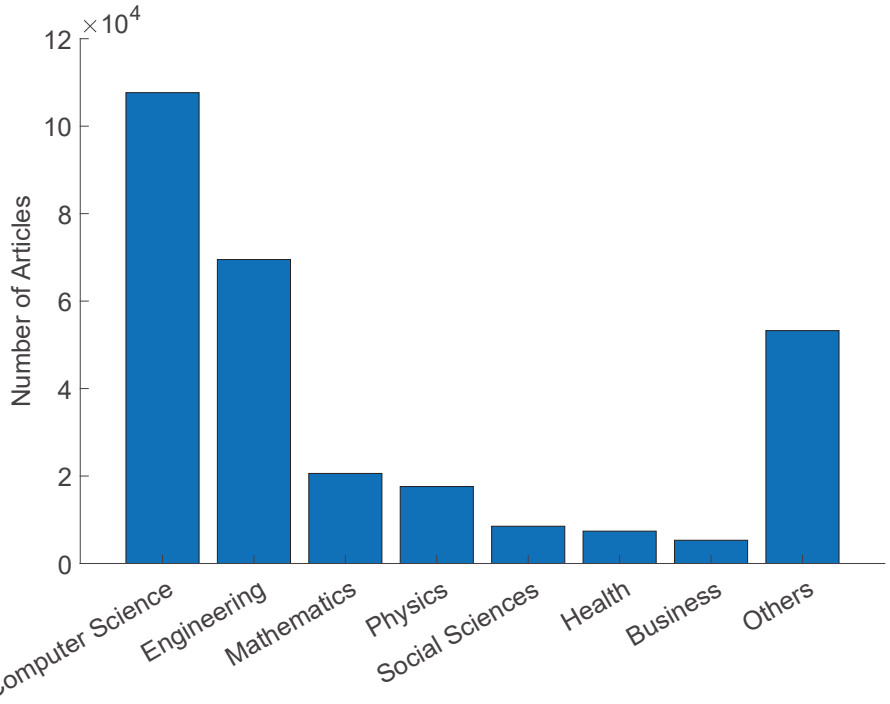

Figure 1.2: Scopus research on IoT: number of published articles per subject area.

the Computer Science and Engineering subject areas. It is also interesting to underline that in the "others" category, added for typographical reasons, it is possible to find very different subject areas like arts, chemistry, psychology and energy. By giving a look to some articles found in this research, and analyzing the graphs above it is possible to understand that IoT is really spreading in every human field. For example, authors of [5] present an IoT-based environmental monitoring system, devoted to the preservation of relics. Moreover, in [6] an IoT system for fire safety purposes is discussed. Agriculture [7], smart buildings and cities [8]–[10] and telemedicine [11], [12] are also other important application fields. For these reason, a common practice nowadays is to define *IoXT* acronyms, to identify different IoT applications. For example, Internet of Wearable Things (IoWT) [13] refers to all those kind of IoT systems

that can monitor health parameters and are wearable. Another famous example is the so-called Internet of Vehicles (IoV) [14], that allow to enact the transition from the classic VEhicle ad-hoc NETworks (VANETs) to novel distributed systems. Following this trend, in the following I will refer to Internet of Measurements (IoM) to represent IoT - based distributed measurements systems, whose characteristics are discussed in the following sections.

It is of fundamental importance to introduce the architecture of an IoT system. It can be derived from the enormous amount of articles in the field. Despite this, a general architecture has been figured out by authors of [15]. In particular, they identified 3 different levels of the IoT architecture, each of them grouping various technologies. A graphical representation of the IoT architecture, derived following [15], is given in Figure 1.3.

As a matter of fact, the activity of an IoT system is done by several devices, the *objects*, capable to actively interact with the environment. In practice, each device is equipped with some **sensors** and / or **actuators**. Moreover, each device has the capability to interact with the other devices, thus being equipped with a communication module. In the last years, IoT devices are becoming increasingly cheaper and simpler to use. Think about how simple can be building a smart home now by using, for example, Alexa. It is possible to find a lot of smart devices that can be connected to Alexa, from the light bulb to the coffee machine. With sensors (microphones for example) and actuators (speakers for example) it is possible to simply interact with the surrounding environment and the user. Afdterwards, a second level could be identified as the network level. Here the communication between the smart devices and the upper layer is managed, for example by using Gateways. Data are then communicated to a data plane, a cloud, where post-processing activity can take place. From this level, the interaction with the user, depending on the specific application, can take place. As already stated before, the main enabling technologies of the IoM are surely the communication protocols (mostly wireless ones). As we will deeply discuss in Section 1.2, communication has an important impact on the performances of the IoM system. Moreover, novel ML, and generally speaking, AI algorithms are usually considered an enabling technology of IoT. Also in the industrial and I&M fields, several contributions

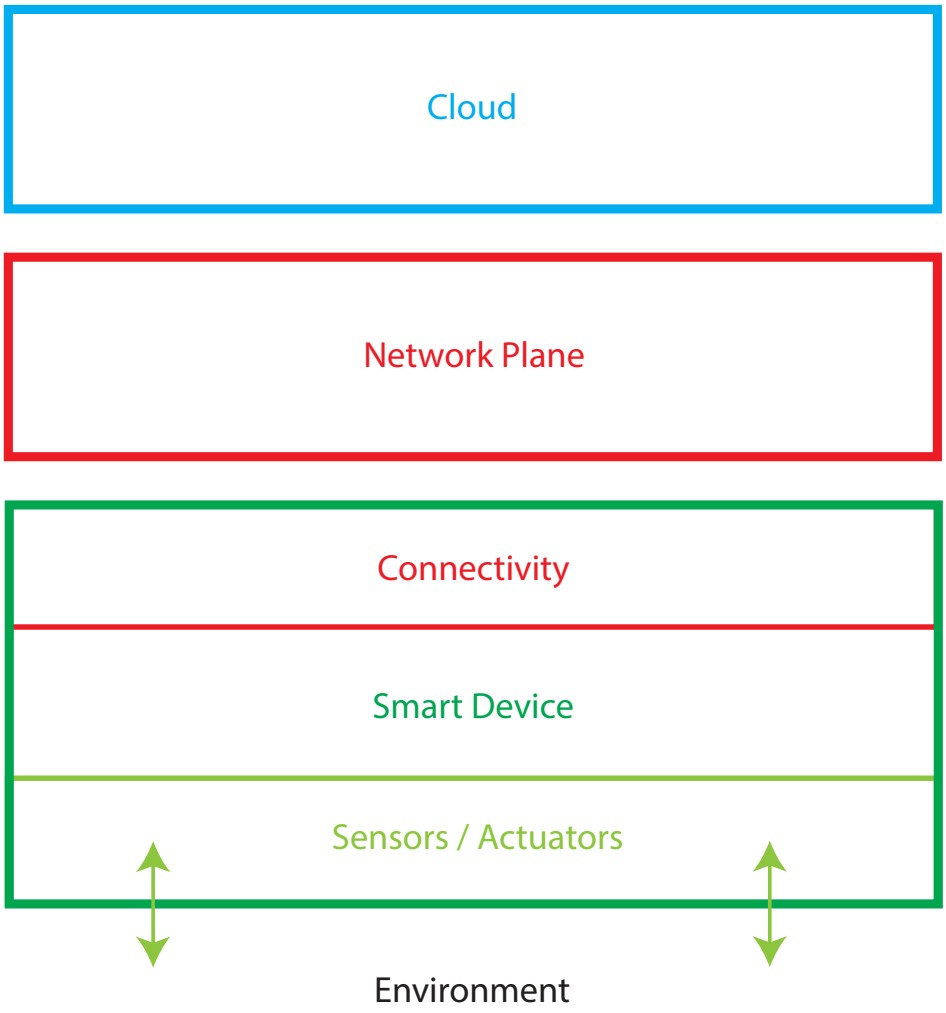

Figure 1.3: Architecture of a general IoT system.

address the topic. One meaningful example are the so-called VBMs, particular IoM applications where measurements are derived from images, i.e. the application of Machine Vision to the instrumentation and measurement field [16]. In this context, Convolutional Neural Networks (CNN)s demonstrated their suitability for VBMs systems in

many different applications [17]. For example, the authors [18] proposed a CNN-based system for railway network inspection. Differently, [19] presented a method for head pose estimation in vehicles. Furthermore, an interesting review paper [20] summarizes the most widespread Deep Learning (DL) techniques for Image Classification, with regard to ophthalmic applications. Despite this, the topic is still challenging as the trustworthiness of AI and ML must be investigated [21].

1.2 Requirements for distributed measurements systems

In previous Section, the IoT concept has been introduced, by analyzing both the main application areas and the general architecture. Both IIoT and IoM concepts have been introduced, and in the following I will refer indistinctly to IIoT and IoM. Indeed, essentially the requirements coming from both are the same, and it is possible to say, in substance, that Industrial IoT systems are one of the possible applications of IoM. In this section, requirements in the field are deeply analyzed.

Networks employed in real–time industrial measurement systems have been extensively addressed by the scientific literature [22], [23]. They are communication systems specifically designed to effectively interconnect industrial devices such as controllers, sensors and actuators. This type of communication implies the timely transmission of limited amounts of data (some tens of bytes, typically) carrying, for example, set–points values or process data variables. Due to the peculiarity of the applications for which they are deployed, industrial networks have often to comply with tight timing requirements. Indeed, they have to grant for the precise scheduling of periodic messages, as well as to ensure that aperiodic messages are delivered within a–priori specified deadlines. In substance, in the industrial field there is the need for real-time networks. The aforementioned requirements can be generalized, and applied to novel IoT-based measurement systems [3], and applied also for measurement systems in different fields. For this reason, it is possible to generally refer to IoM systems.

In this context, the measurements coming from sensors cover a widespread importance, impacting on several data flows. In particular, on cyclic real-time flows, alarms

and events. In this context, sensors data need to guarantee certain levels of measurement precision and accuracy, thus allowing to suitably handle critical situations and / or stably controlling a process. Furthermore, in the harsh industrial environment, it is of fundamental importance the analysis of the impact of complex, distributed and IIoT measurement networks, even wirelessly connected, on the measurement accuracy. In particular, there is the need for a precise analysis (by using also new measurement metrics) of the impact of such new intelligent and smart systems on the measurement activity. This problem is even more critical if the measurement system uses AI techniques, or vision systems, the latter dramatically increasing the amount of time–critical data to send and process. In this context, a significant example is the impact of the transmission delay on the measurement process. Indeed, assuming that at a specific instant of time t_s the sensor sends a measure x_s, the data will be received at an instant of time $t_r = t_s + t_d$, where t_d is a random variable describing the delay introduced by the communication network. For this reason, the communication network has an impact on the measurement uncertainty, as the real value of the measured variable is $x_r \neq x_s$.

From a measurement point of view, this issue seems to present different simple solutions, especially if the t_d uncertainty can be neglected. Indeed, considering only the measurement aspects, it is both possible to timestamp the data coming from sensors or adjust the deterministic error after data reception. Obviously, this kind of solutions can be applied only if the analysis is carried offline, and measurements do not need to be used in real-time. Roughly speaking, if a measurement is needed before a certain instant of time, if it arrives later, even timestamped, it is not useful anymore. Unfortunately, in an Industrial scenario measurements need to be used in real–time to control a process or handle critical situations, such as alarms or sporadic events. In this context, several works focus on the possibility, from a control perspective, to model and take into account the network delay in the control design stage. For example, authors of [24] try to compensate both network delay and packet loss by suitably designing the control stage. For example, the delay could be taken into account by using a e^{-st_d} term in the control model.

In this context, the problem is that network delay is not even deterministic, as in general t_d follows a specific probability function. From a measurement point of

view, according to the International Organization for Standardization (ISO) Guide to the Expression of Uncertainty in Measurement (GUM) [25], it is possible to evaluate the uncertainty (type B) introduced by the communication network delay t_d, as per Eq. (1.1).

$$u(x) = \left| \frac{\delta x(t, t_d)}{\delta t_d} \right| \cdot u(t_d) \tag{1.1}$$

Where x is the signal received from the sensor, that depends on both t and t_d, and $u(t_d)$ is the uncertainty on the knowledge of t_d. From the latter observation, it is possible to conclude that lowering $u(t_d)$, by using a deterministic network, lowers the measurement uncertainty. In particular, measurement data need to be handled with a certain priority, given by the critical level of the specific operation, that in turn reflects on the measurement uncertainty. It is worth observing that, practically, the calculation of $\frac{\delta x(t,t_d)}{\delta t_d}$ can be approximated by evaluating the dynamics of the specific sensors employed, thus deriving the $\frac{\Delta x}{\Delta t}$ of the sensor. This is possible as $x(t, t_d) = x(t - t_d)$, thus involving in $\left|\frac{\delta x(t,t_d)}{\delta t_d}\right| = \left|\frac{\delta x(t,t_d)}{\delta t}\right|$. If the measurement system has been well designed, the sensor dynamics needs to be fast enough to capture the measurand variations, thus being the latter approach a worst-case analysis.

The widespread used communication networks for industrial applications, namely fieldbuses and Real Time Ethernet (RTE) networks (Section 3.1) are applicable to handle the requirements coming from the Industry 4.0 paradigm in terms of timeliness and real-time behavior. This topic becomes critical considering the need for IoT smart measurement systems, and wireless connectivity, as underlined also by the Physikalisch-Technische Bundesanstalt (PTB) [26]. Notably, packet transmission periods and deadlines can be as low as some hundreds of microseconds. The introduction of novel technologies, such as wireless networks in this challenging context needed careful analyses, due to the well known problems that may affect wireless communication such as fading, path loss and collisions, that inevitably lead to delays in packet delivery or, even worse, to packet losses [27], [28]. For this reason, both the communication

delays and the so-called Data Extraction Rate (DER), defined in Eq. (1.2), must be monitored to understand if a specific network protocol is tailored for IoM or not.

$$DER_t = \frac{RP_t}{SP_t} \cdot 100 \quad (\%) \tag{1.2}$$

Actually, SP_t and RP_t are, respectively, the sent and the correctly received packets before the specific instant of time t.

This topic has been addressed by several research activities in the past years, that led to the latest state of play in which wireless networks are widespread in the industrial scenario, particularly for process automation applications [29], [30]. Moreover, their deployment is expected to grow significantly in the next future [31]. It is worth observing that, to accurately handle time-critical traffic, synchronization between the different devices is of fundamental importance. For this reason, both traditional industrial networks (Section 3.1) and novel standards like TSN (Section 3.3) design accurate syncronization algorithms.

As a last consideration, within the IoM and IIoT fields, the IoT architecture, already discussed in Section 1.1, must also be optimized. Indeed, the IoM systems can also benefit of cloud computing, that can be exploited to improve the reliability, scalability, interoperability of the CPS [32], as already pointed out in Section 1.1, providing a centralized asset devoted to enhancing data storage, computing capabilities. However, while the cloud computing framework may foster the relationship between the physical and the cyber (computing & control infrastructures) part of the system, in the illustration of Figure 1.4 some of the critical issues undermining its implementation are summarized [32], [33].

Principally, unbounded latency and delays yield a non-deterministic system, unable to handle time-critical or real-time tasks. Moreover, the big amount of data can overload the network infrastructure, given the centralized nature of cloud servers, which in turn may lead to serious security problems. A possible solution is found in the edge computing paradigm, which demands for the availability of general-purpose devices, with own computational and storage resources, at the edge of the system [34], [35]. Multiple edge devices are placed next to the physical entities realizing a computa-

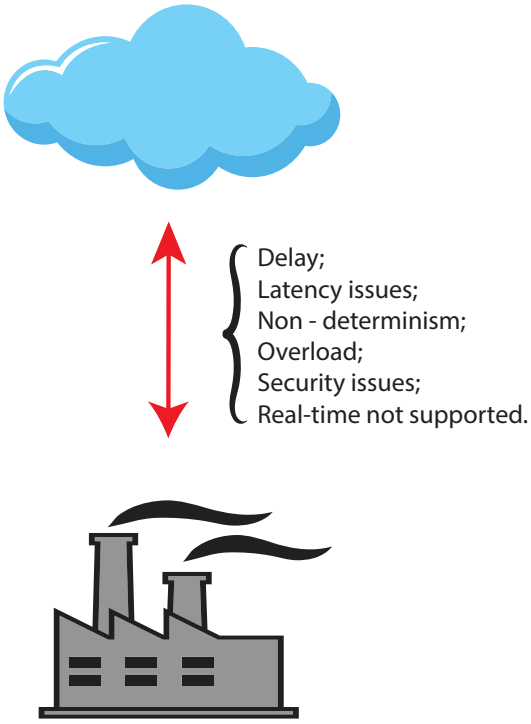

Figure 1.4: Limitation on the usage of cloud computing in the I&M scenario.

tional offload from the cloud to the edge devices, and hence effectively serve a lower number of sensors. This strategy may help achieving an effective reduction of the latency and delays associated to centralized computation and communications, as well as may significantly increase robustness, safety and security. Even more, a significant breakthrough can be provided by the development of what is called fog computing, a novel decentralized architecture which is able to meet all the requirements of modern smart factories [36], in terms of ultra-low latencies, reliability, Quality of Service (QoS), security and functional safety.

As a matter of fact, the IoM can be developed by using edge or fog computing, in order to provide reliable and real time measurement communication and process. In the following, as already stated before, the work will focus on the optimization of wireless

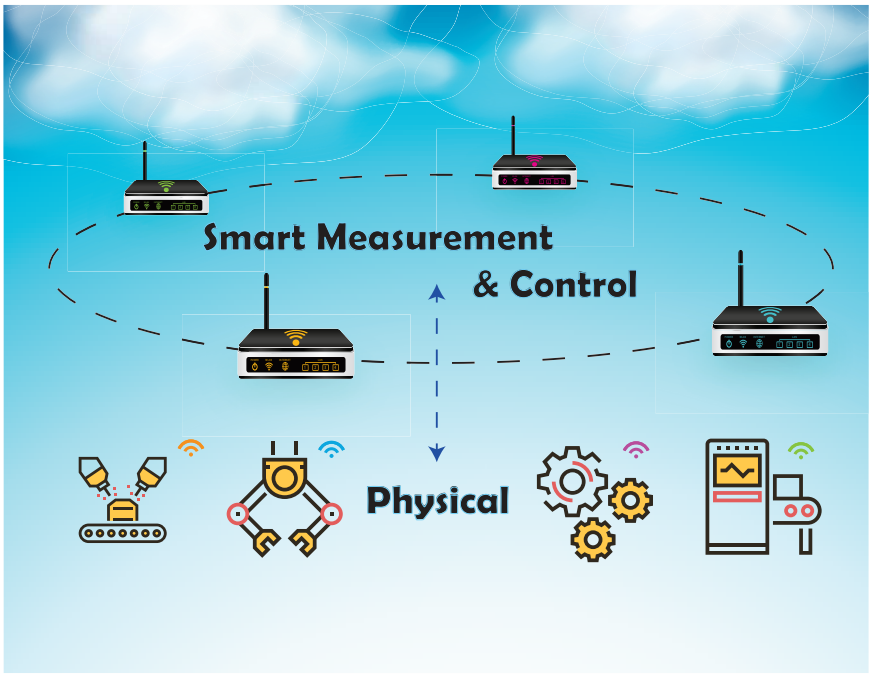

Figure 1.5: Edge computing representation.

protocol for the IoM field, whose architecture foresees the usage of intermediate edge and/or fog layers, as depicted in Figure 1.5.

In the following, this architecture is not studied in detail, as the focus will be on the communication between sensors and the devices placed next to the field, aiming at controlling a process or monitoring a phenomenon. Moreover, several ML techniques will be exploited to both optimize the communication and for post-processing purposes.

As already stated, between all the application fields of the IoT, the Industrial one, and in general IoM, is certainly gaining much research interest in the last years. The research interest on the IIoT is demonstrated also by Figure 1.6.

A similar trend can be noticed when searching IoT in the sources (conferences and journals) in the I&M field, as can be seen in Figure 1.7.

As a matter of fact, the research on IoT systems in the I&M field started some years later, but the interest on IoM is still growing. Despite this, the research on this

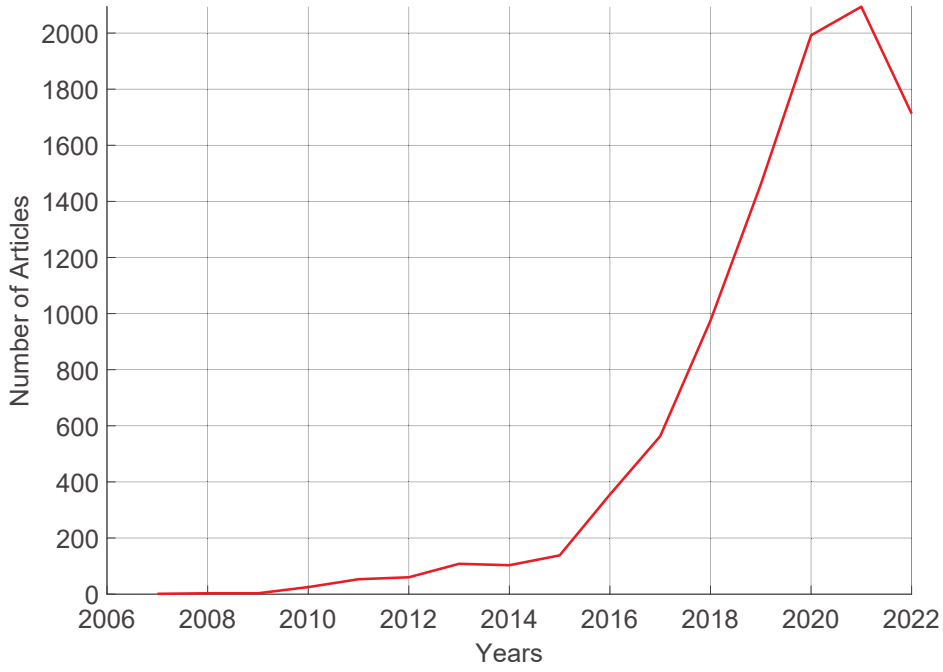

Figure 1.6: Scopus research on IIoT: number of published articles per year.

field is still limited (only approximately the 3% of the total research on IoT).

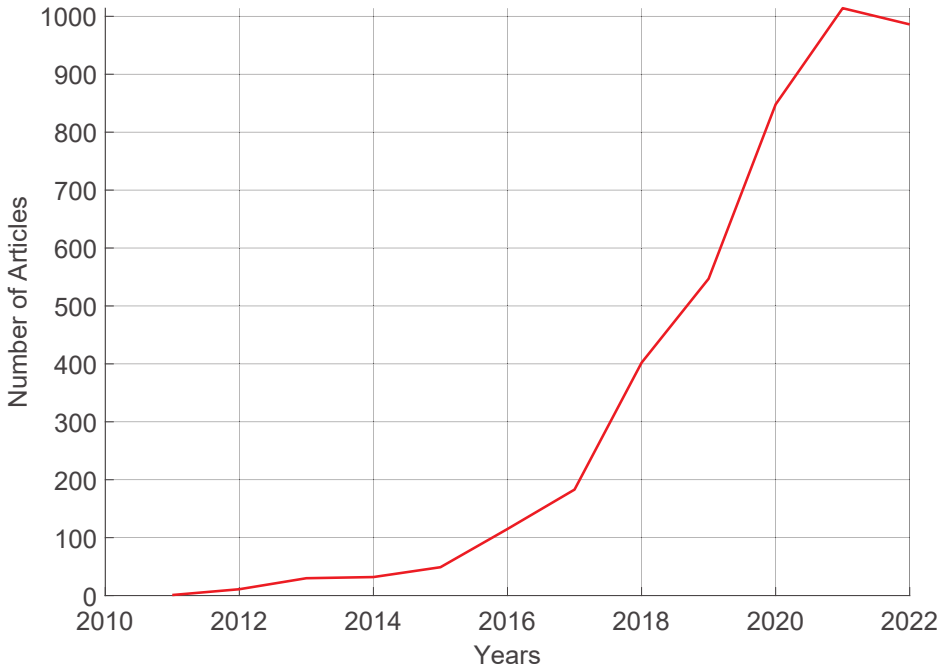

Figure 1.7: Scopus research on IoT in the I&M field: number of published articles per year.

Chapter 2

An Introduction to AI and ML

"I believe there is no deep difference between what can be achieved by a biological brain and what can be achieved by a computer. It, therefore, follows that computers can, in theory, emulate human intelligence - and exceed it.[1]*"*

The AI is one of the most important and discussed topics on the last years. In practice, AI groups a plethora of different techniques aiming at emulating the human intelligence, thus allowing to smartly solve some challenging topics, or even to create intelligent machines. Such an approach, rather then hardly coding instructions, proved to be particularly efficient in a series of different applications such as health [37], agriculture [38], transportation [39] and even industrial [40]. Despite this, both insiders and non-insiders are still discussing about ethical issues and possible future negative aspects of AI.

In this chapter, I will go through general definitions in the AI panorama and we will deeply analyze the most used AI techniques in the I&M field.

[1] Stephen Hawking

2.1 The Artificial Intelligence concept

Developing increasingly intelligent machines is surely a key topic in the next years. But how it is possible to suitably define AI? The discussion about the most precise definition of AI dates back in the '50s, together with the development of first computers. In particular, Alan Turing proposed the so-called *Turing Test* [41], aiming at answering to the question: *"Can machines think, like humans?"* The Turing test, or the so-called *Imitation Game*, is described by Theorem 2.1.1.

Theorem 2.1.1. *The Imitation Game*

Consider one computer, A, a human B and a human interrogator C, the latter being in a separate room with the respect to the previous ones. The interrogator can ask questions and receive answers from both A and B, via a suitable system of keyboards and screens. In this game, the computer tries to trick the interrogator, by answering to the questions like a human being will do.

If the interrogator is not able to identify who is the machine and who is the human, the machine can be considered intelligent.

□

One of the most famous critic to the Turing test comes 30 years later, in 1980, from John R. Searle [42]. In practice, he underlined that a Machine could easily pass a Turing test by learning a natural language, without any consciousness or real understanding of it. In this context, it is also normal asking ourselves if a machine able to *act* as a human, is also able to *think* as a human. As a matter of fact, as pointed out also by authors of [43] the problem is not to define what *Artificial* means, but what *Intelligence* means, especially if applied to inanimate entities, like machines. Indeed, the concept of *Intelligence* is generally and intuitively related to a plethora of human capabilities like the possibility to react to specific situations, to take actions in a specific environment, to learn from experience, the intentionality and the rational behavior, only to name a few. The possibility to a-priori label a technique as "artificial intelligence" is still challenging, because it also depends by the specific definition of an agent, capable

to take rational and intentional actions in specific environment, etc. For this reason, practically, we will refer in the following to several established AI techniques, trying to implement some of the aforementioned human capabilities. As a matter of fact, [44] considers four different definitions of AI:

1. Thinking Humanly;
2. Thinking Rationally;
3. Acting Humanly;
4. Acting Rationally.

The detailed description of each formulation is actually out of the scope of this work. Despite this, it is worth observing that, talking about *intelligence* acting and thinking are two different features. Moreover, authors split the rational acting or thinking from the human acting and thinking. This is because the strict *rational* context can be seen as the capacity to achieve the most reasonable solution of a problem. In this context, human intelligence is not related always with the best result, because depending on the specific problem, each human can simply make mistakes. Moreover, intelligence is also intentionality, such as a human can also, in principle, intentionally decide not to choose the way to the best result.

In general, an intelligent machine relies on the concept of *Agent*. The agent runs the so – called *agent program*, implementing an *agent function* that allows to act on the environment basing on some inputs data derived from the environment too. The general scheme is depicted in Figure 2.1.

Besides the definition of AI is still object of discussion between insiders, it is possible to conclude that an Agent is said to be rational if it takes the best action in a specific moment [44]. As a matter of fact, the process relies in a perception and action cycle, where only the whole sequence of data acquired from the environment in the past can affect the action taken in each moment. Now it is possible to understand why measurement systems can benefit from AI. Indeed, imagine an IoT platform, collecting data to be used as direct feedback for a control loop or to further analysis. The IoT

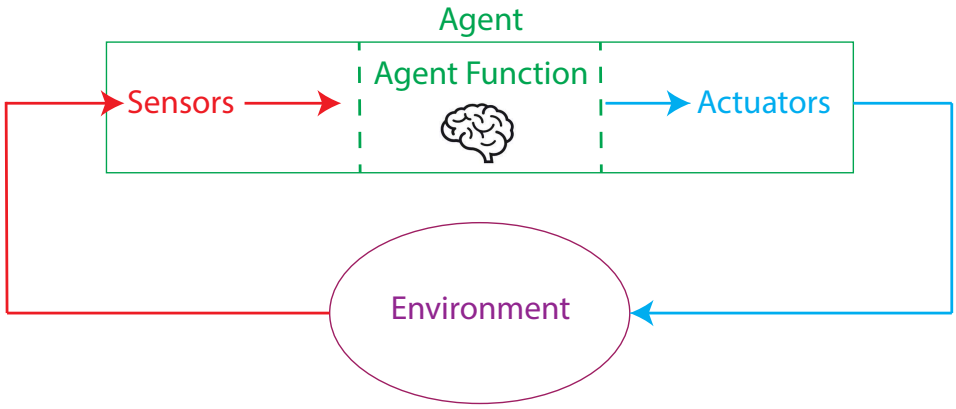

Figure 2.1: Concept of Intelligent Agent.

device can run a suitable *agent program*, to properly process and analyze the collected data.

2.2 Typologies of AI algorithms

In Figure 2.2 are depicted the possible subsets of AI algorithms.

As it is possible to see, there is a big difference between AI and ML, the latter being a subset of the former. In particular, ML algorithms are essentially one possible *engine* of an AI algorithm, from which AI can learn and improve with experience. As a matter of fact, ML is not the only way an AI algorithm can work. Figure 2.3 represents the typologies of ML algorithms.

In particular, ML algorithms can be grouped into **supervised**, **unsupervised** and RL ones. Supervised learning groups a set of algorithms whose training activity is carried out by using a set of desired output. In this context, the training dataset must contain both inputs and desired outputs. As a matter of fact, two different activities can be carried out by supervised algorithms, namely classification and regression. The former deals with data classification, where the algorithm's aim is to label the input data, to divide them in different classes. On the other hand, regression algorithms are

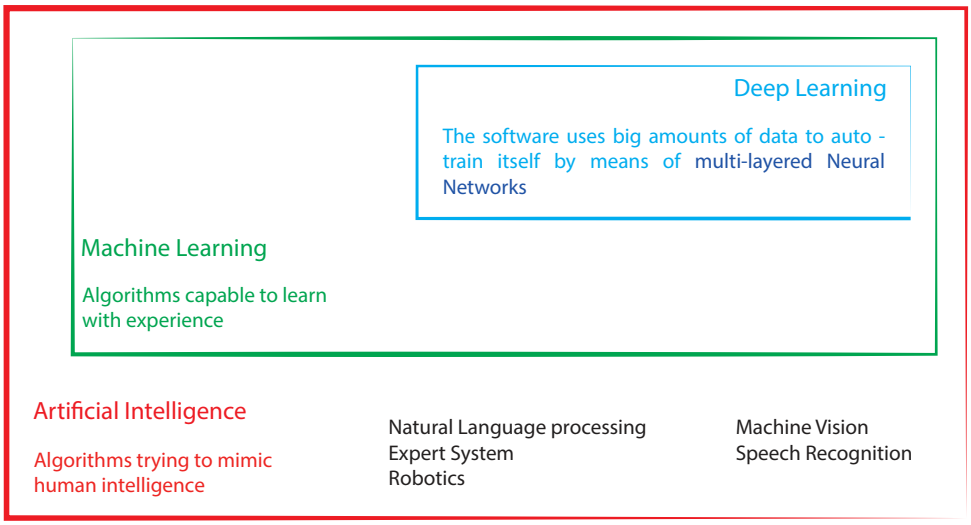

Figure 2.2: Artificial Intelligence algorithms and difference between AI and ML

capable of estimating values of a data series, from a set of known samples.

Some well-known supervised ML algorithms are listed below.

1. k-Nearest Neighbours
2. Linear Regression
3. Logistic Regression
4. Support Vector Machine (SVM)
5. Decision Tree
6. Random Forest (RF)
7. Neural Network (NN)[2].

Unsupervised learning techniques, differently from the previous ones, train directly from the input data, without the need for desired output data. These algorithms are

[2]Neural Networks can also be unsupervised

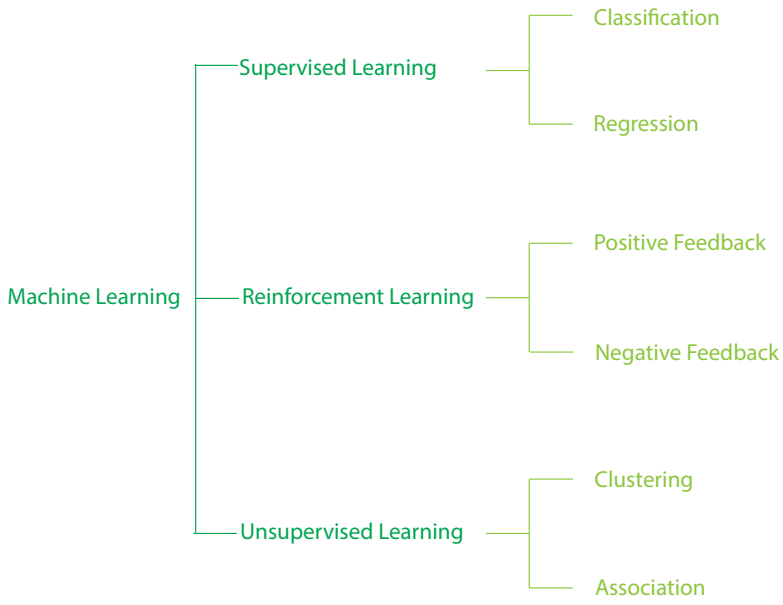

Figure 2.3: Machine Learning algorithms classification

capable to automatically identify the characteristics, the patterns (namely features) of the input data.

Unsupervised learning algorithms can be divided into two different classes.

1. Clustering Algorithms;
2. Associative Algorithms.

The first ones are, in substance, the unsupervised counterpart of the classification algorithms. Indeed, input data can be divided into different groups (namely clusters) basing on some similarity patterns.

Secondly, associative algorithms can identify relationships between variables, features in huge datasets. The latter are mostly used for marketing purposes, such as for predicting the incomes of some products. Another important activity is the so-called

dimensionality reduction. Sometimes the input dataset is extremely big, with several unuseful features. In this context, it is really important to identify the most important characteristics and / or recombine input data. In this context, correlation between features must be analysed or some feature extraction algorithms can be used, such as the Principal Component Analysis (PCA).

In the following of this chapter, several ML techniques will be analysed, particularly focusing on the most used in the I&M field and the ones used on the following chapters. In particular, linear regression (Section 2.3) and logistic regression (Section 2.4) are investigated in order to better analyse, in Section 2.5, the Neural Networks. The SVM are investigated in Section 2.6, RF in Section 2.7. The unsupervised K-Means clustering algorithm is investigated in Section 2.8 and the RL in Section 2.9. Finally, Sections 2.10 and 2.11 are going to point out, respectively, the metrics used to evaluate classification algorithms and the training process of a classification supervised NN. The latter sections will be particularly useful in the following, to better analyse the experimental results. In this work, Python (in particular TensorFlow, Keras and Sci-Kit Learn Python frameworks) is used to implement the various ML techniques. In this chapter, to introduce the various techniques and the metrics part, the widespread fashion mnist dataset is used as an example.

2.3 Linear Regression

As said before, regression algorithms aim is to predict values of a data sequence, from a limited number of samples of the sequence. If we imagine to have as inputs a vector of n inputs, $x_i | i \in \{1, \dots, n\}$, according to the linear regression algorithm the output is generated as the weighted sum of the inputs, plus a so-called bias (θ_0) value, as per Eq. (2.1).

$$\hat{y} = \theta_0 + \theta_1 \cdot x_1 + \cdots + \theta_n \cdot x_n \tag{2.1}$$

Generally speaking, supervised algorithms foresee the tuning of some parameters, basing of the inputs and the known outputs. Linear regression parameters are $\theta_i | i \in \{1, \dots, n\}$, and are tuned in order to achieve the best linear fit possible of the known

data. For this reason, a common practice in all the supervised learning algorithms is the definition of a suitable *loss function*. In particular, the most simple way to evaluate the prediction error is by calculating a Mean Error (ME), as per Eq. (2.2).

$$ME = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i) \tag{2.2}$$

Obviously, in this way the negative errors decrease the mean error. For this reason, an Absolute Mean Error (AME) can be calculated, as per Eq. (2.3).

$$AME = \frac{1}{n}\sum_{i=1}^{n}(|\hat{y}_i - y_i|) \tag{2.3}$$

Another possibility is to calculate a Root Mean Squared Error (RMSE), as per Eq. (2.4)).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{2.4}$$

The RMSE is probably the most used loss function, as it weights more big prediction errors. The RMSE must be used without considering outliers, that negatively affect the error calculation.

The linear regression can be simply implemented in Python, by using Scikit - Learn. An example is depicted in Figure 2.4.

As a matter of fact, the algorithms automatically adjust both the slope and the intercept (this is why the bias value θ_0 is needed) of the line, trying to achieve a best fitting line. From the derived line, it is possible to estimate the unknown data. The evaluation can be carried out by one of the errors introduced before, or with the so–called determination coefficient ($0 \leq R^2 \leq 1$), where 0 means "wrong prediction" and 1 represents "perfectly predicted". It is worth noting that the linear prediction is not much accurate when data have not a linear shape, as can be seen when comparing Figures 2.4 and 2.5.

Usually, it is possible to consider a small portion of the whole data, that can be considered linear. Moreover, this algorithm become the starting point for the following

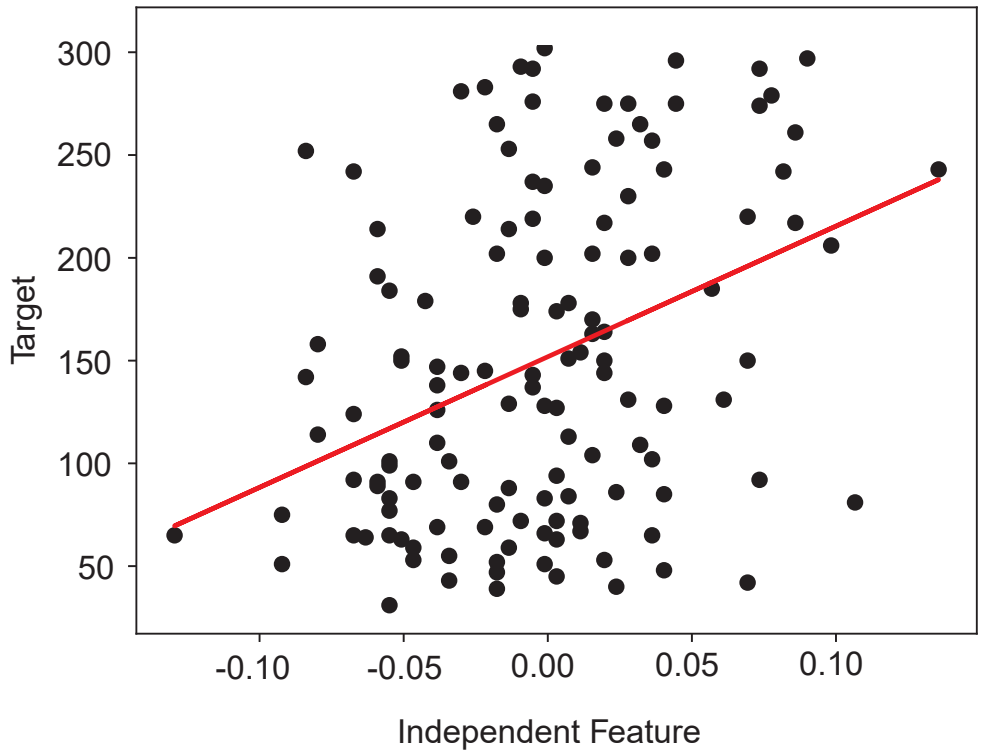

Figure 2.4: Example of Linear Regression. MSE = 5594, $R^2 = 0.1155$.

ones, and even of the famous perceptron, from which the Neural Networks can be built.

2.4 Logistic Regression

The Logistic Regression is a classification supervised learning algorithm. In substance, Logistic Regression works exactly as the Linear one, introduced in Section 2.3 and it adds a sigmoid function after the calculations of Eq. (2.1).

As a matter of fact Eq. (2.1) becomes:

$$\hat{y} = \sigma(\theta_0 + \theta_1 \cdot x_1 + \cdots + \theta_n \cdot x_n) \tag{2.5}$$

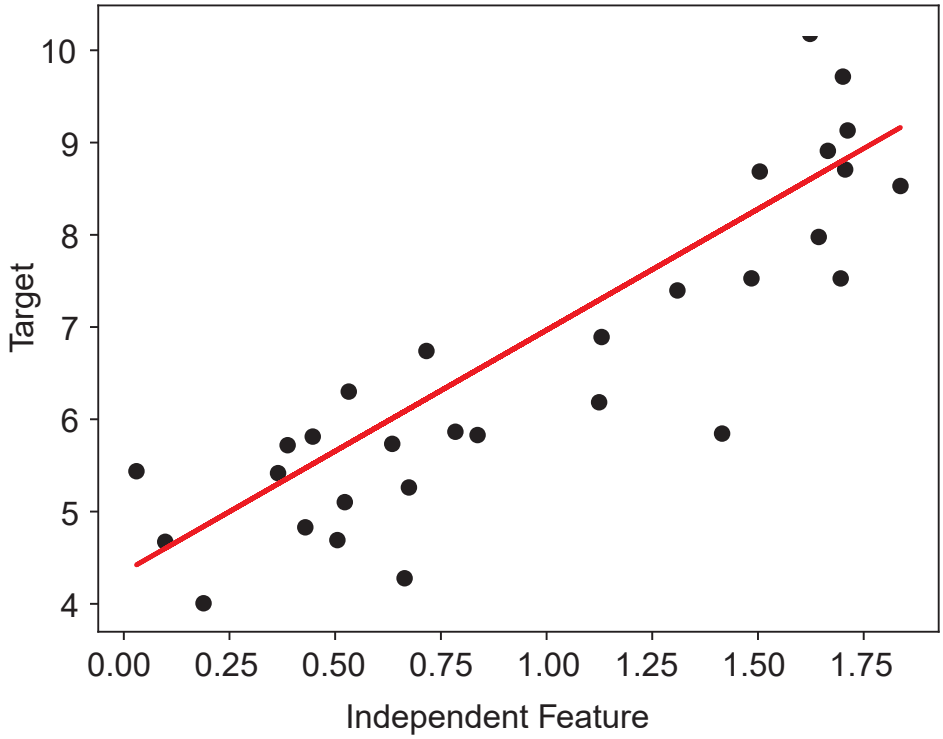

Figure 2.5: Example of Linear Regression (2). MSE = 0.586458650226466, R^2 = 0.85.

Where:

$$\sigma(t) = \frac{1}{1 + e^{(-t)}} \tag{2.6}$$

Whose graph is depicted in Figure 2.6.

As the logistic regression gives as an output a value between 0 e 1, the logistic regression is particularly suitable to return the probability that an example belongs to a particular class, thus being a classification algorithm.

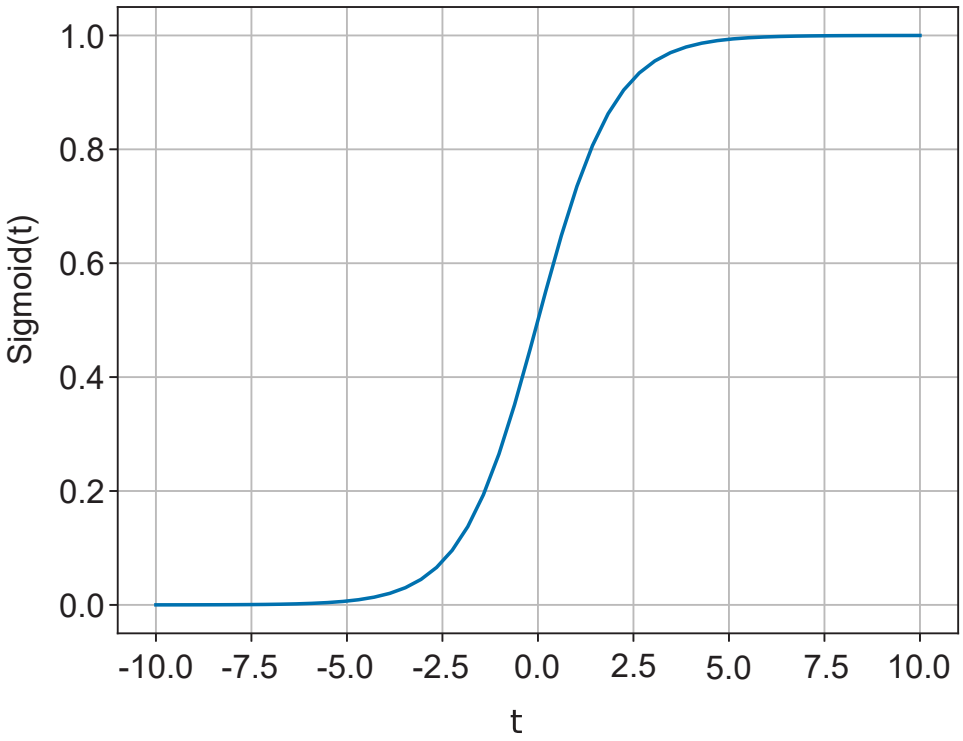

Figure 2.6: Graph of a Sigmoid function.

2.5 Neural Networks

NNs are surely one of the most used ML algorithms, and can be used to solve several different problems, such as classification or prediction. NN can be both supervised and unsupervised. In this section, NNs are introduced, specifically referring to supervised classification problems, as in the following chapters classification problems using NN are addressed.

A NN is a ML algorithm that foresee to mimic the behavior of the human brain. The basic element of a NN is the so-called perceptron, that is represented in Figure 2.7. The perceptron can be considered the artificial counterpart of the human neuron.

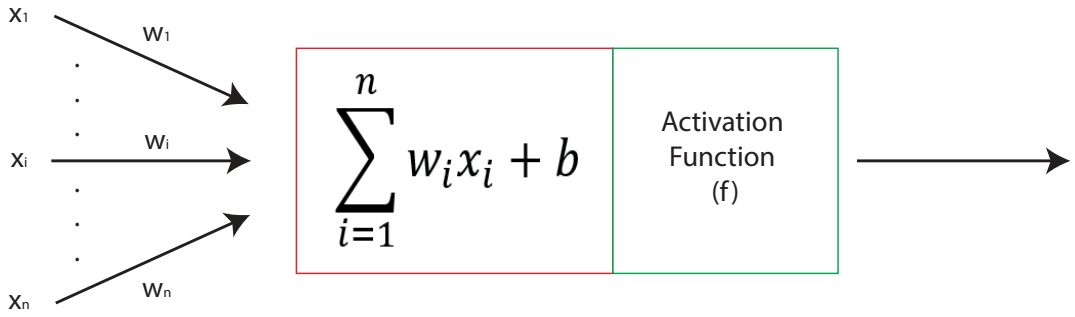

Figure 2.7: The perceptron.

The single perceptron, as can be seen from Figure 2.7, does the same operations explained in the previous sections for the linear and logistic regressions. In particular, given a series of inputs $x_i | i \in 1, \cdots, n$, the output value is calculated as per Eq. (2.7).

$$y = \sum_{i=1}^{n} (w_i \cdot x_i) + b \tag{2.7}$$

Where $w_i | i \in 1, \cdots, n$ are the weights of the perceptron, and b is the boas value.

As a matter of fact, the NN is organized into layers of perceptrons, that are properly interconnected between them. In practice, a single perceptron can be considered the most simple classifier. It draws a line to separate a group of data, in two different classes and then passes the output to an activation function. The latter can be used, in classification problems, to give as an output a probability value and one of the possible activation functions can be the already introduced sigmoid (Eq. (2.5)). In substance, a single perceptron can solve linear binary classification problems, where data are linearly separable. For more complex problems, group of perceptrons working together muct be employed, giving rise to the NN. A possible representation of a NN is presented in Figure 2.8.

These networks are usually called Multi Layered Perceptron Neural Networks (MLPNN), and are characterized by one input layer, one output layer and one or more hidden layers between the former two. In general, networks with a huge number of hidden layers between input and output, trained with a huge amount of data, are

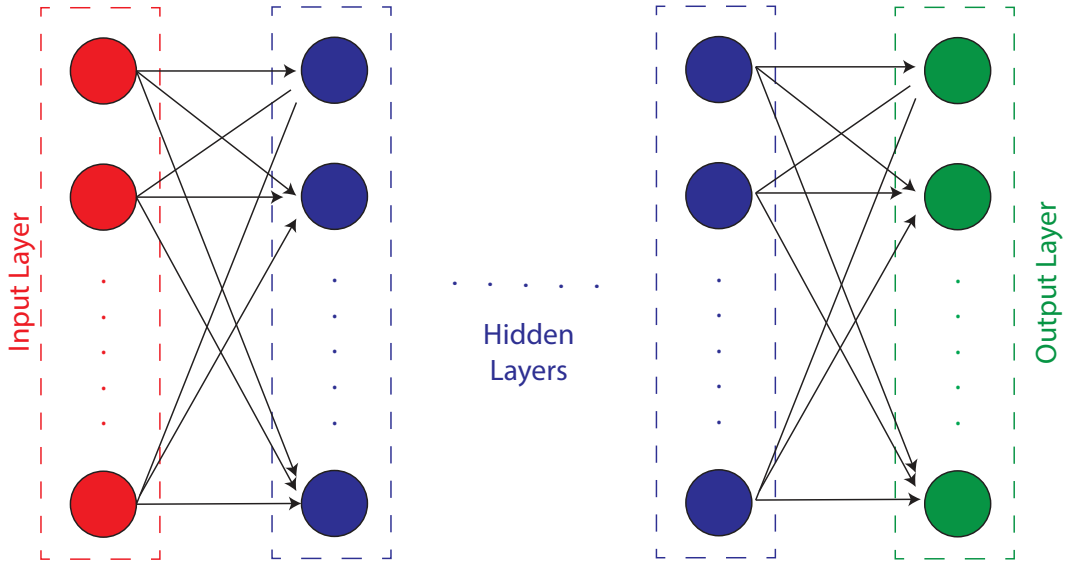

Figure 2.8: Neural Network example.

characterized by a so-called DL.

2.5.1 Training process

The fundamental components of the training process of a NN are depicted in Figure 2.9. In particular, the network initially sets random values of the weights in each neuron. At each training step, the network generates an output value (that in a classification problem is a vector containing, at each index i, the probability that the specific input belongs to the class i), namely forward process. A specific **Loss Function** calculates an error between the prediction and the real value, that in turn can be used from an **optimization algorithm** to modify the weights in order to minimize the estimation error. In this context, the so-called **backpropagation** is the process taking place from the output layer (where the error is generated) through the hidden layers (where the weights are modified) to the input one.

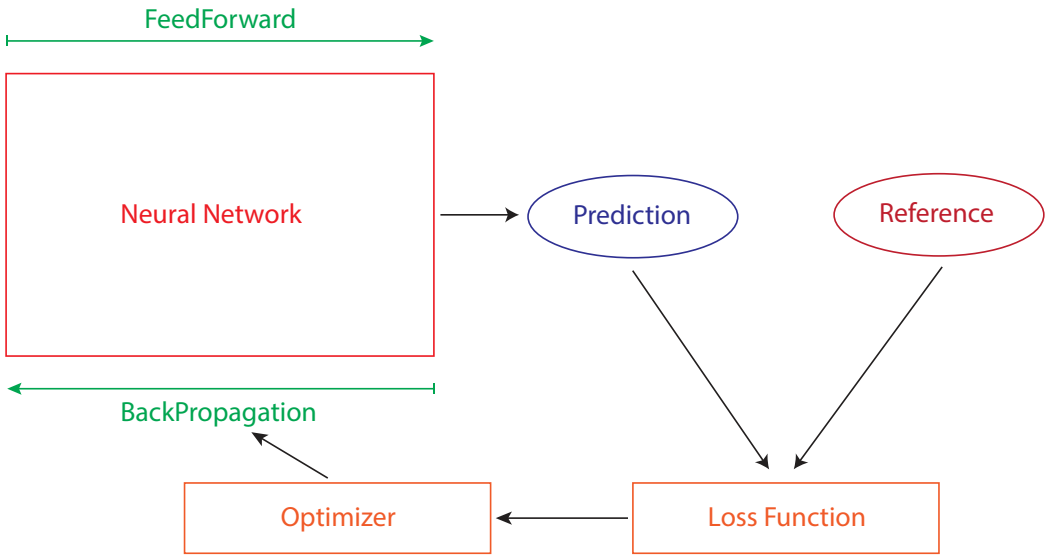

Figure 2.9: NN basic elements.

2.5.2 Convolutional Layers

Complex image classification problems can be solved also by using the so-called convolutional layers. A general example can be found in Figure 2.10, and they foresee to use a suitable kernel, that is essentially a moving filter that scans the whole input figure, trying to give more importance to specific features, like borders for example. In this context, single layers values are multiplied by the kernel ones.

Usually CNNs are formed by both Convolutional and Pooling layers, where the latter are placed between each convolutional layer and the subsequent one. Specific image patterns are identified by means of a Kernel slicing on the entire layer. Afterwards, a common practice consists in the usage of a pooling layer, that downsamples the input patterns, aiming to increase the robustness of the network to slight variations of the detected features. The extraction of such meaningful features is done taking the maximum from each kernel acquisition, namely *max pooling*. Finally, the last fully connected layers and the *softmax activation* compute the probability of each image to

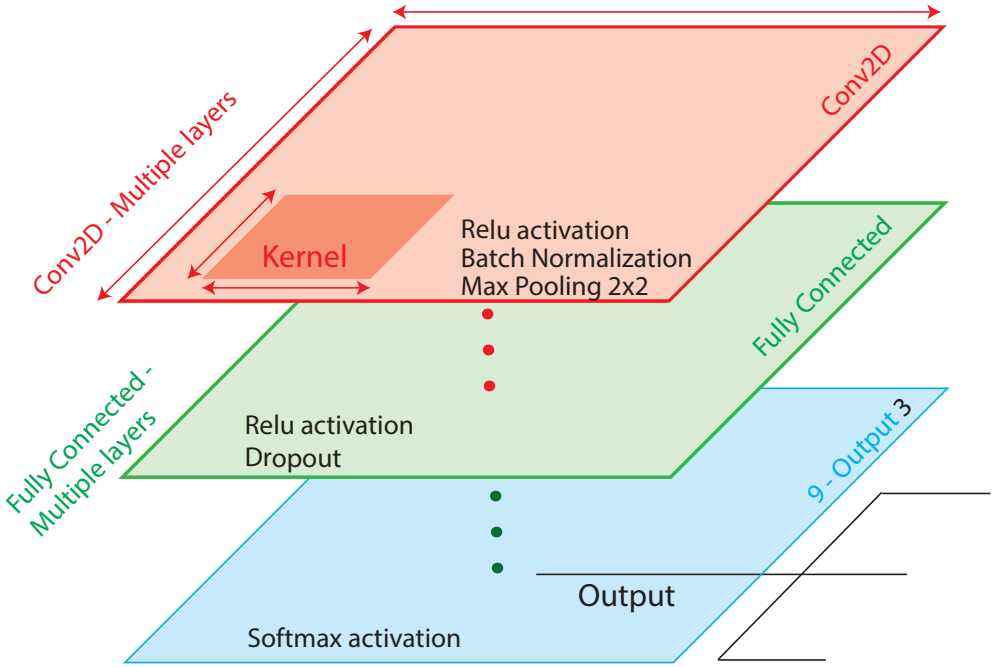

Figure 2.10: Example of a typical Convolutional Network. There is a convolutional step, and a subsequent classical classification.

belong to each of the 3 classes.

These networks are gaining grater importance in the image classification context. Indeed, an interesting review paper [20] summarizes the most widespread Deep Learning (DL) techniques for Image Classification, with regard to ophthalmic applications. They identified several CNN network models that proved to be promising such as, among others, AlexNet [45], GoogleNet, [46] and ResNet [47]. These networks showed the best results in the ImageNet Large Scale Visual Recognition Challenge [48] and are all based on Convolutional Layers.

2.6 Support Vector Machines (SVMs)

SVMs are typically considered a supervised classification algorithm, although if they can be used, in general, also for regression purposes. Generally speaking, a SVM generates a multidimensional plane, namely the Maximum Marginal Hyperplane (MMH), that suitably divides data into different groups. In a two-dimentional plane, the MMH is simply a plane. This plane is found iteratively, trying to minimize a loss function. The plane is defined by the so-called *support vectors*, that comprise the data at the boundary between the different classes. Highest the distance between the support vectors, highest the classification accuracy. The *gamma* value can be tuned, and it is the counterpart of the Learning Rate (LR) in the NN.

2.7 Random Forest (RF)

RF is one of the most used classification ML algorithms. RF bases its behavior on the usage of several decision trees. Decision trees are built by putting in all the internal nodes (i.e. non-leaf ones) a feature of the input dataset, while the leaf nodes are named as the output labels. Each arc of the tree represents one of the possible characteristic of the feature specified by the parent node. In this way, a classification task is simply carried out by starting from the root, following the features, until a leaf node ends the algorithm returning a specific classification.

RF is typically considered a very powerful ML technique, as it employs different decision trees that works in parallel, thus creating a forest of random decision trees. Each tree makes a prediction, and the most voted class becomes the final outcome of the algorithm.

2.8 K-Means

K-means is a clustering algorithm, capable to quickly and efficiently divide a non-labelled dataset into different clusters. The idea is to group the dataset into K different classes, being K a parameter tunable by the programmer. If we imagine to have data

in a plane, and if the goal is to group the data in K different classes, it is possible to firstly identify K centroids. A centroid represents the centre of a cluster, and each data sample belongs to the cluster of the nearer centroid. The K-Means algorithm works as follows:

1. K different centroids are randomly generated.
2. Each iteration implements a double phase technique: expectation-maximization. Firstly, each data sample is assigned to the nearer centroid. Afterwards, the mean squared distance between all the points belonging to a cluster is calculated.
3. K different centroids are generated.
4. Repeat from the step 2.

The quality of the centroids assignments is given by the squared sum of the euclidean distances between the data sample and their centroid. The final centroid assignment is chosen as the one that minimizes such distance.

Practically speaking, the critical choice for the programmer is the number of clusters K. Usually, different values of K are tested and the squared error is taken into account to compare the performances for the different values of K. The typical obtained curve is depicted in Figure 2.11.

This is known as the elbow curve, and usually the chosen K value is the one exactly in the elbow. It represents the Sum of euclidean Squared Errors (SSE) for different K values. As can be seen, while K increases, the error always decreases. This behavior can be intuitively explained: if more centroids are present, more groups are created, resulting in lower distances between data points and centroids. Ideally, by choosing K equals to the number of data points, the error is zero. Despite this, it is possible to notice that in the first part of the curve (before the elbow) the error decreases faster then in the second part. This is because in the first part the error decreases both because of the aforementioned mechanism and really because approaching the optimal K value. Viceversa, on the second part, the error decreases only because of the natural decrease of error in response to a K value increase. To better analyze this behavior

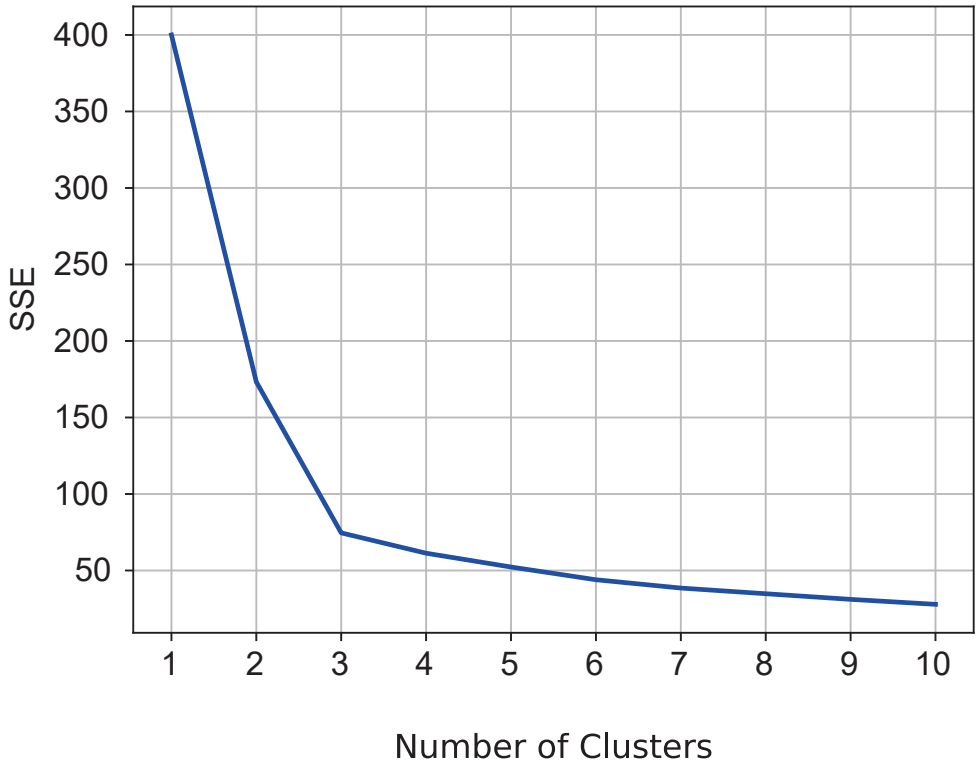

Figure 2.11: KMEans algorithm typical elbow curve.

it is possible to weight the distance between the data points and the centroids with the distance between centroids. In this context, the so-called silhouette coefficients are calculated as per Eq. (2.8).

$$\frac{(CtoC - DtoC)}{max(CtoC, DtoC)} \tag{2.8}$$

Where:

CtoC is the squared distance between centroids;

DtoC is the mean squared distance between data points and centroids.

In the example before, the silhouette values are depicted in Figure 2.12.

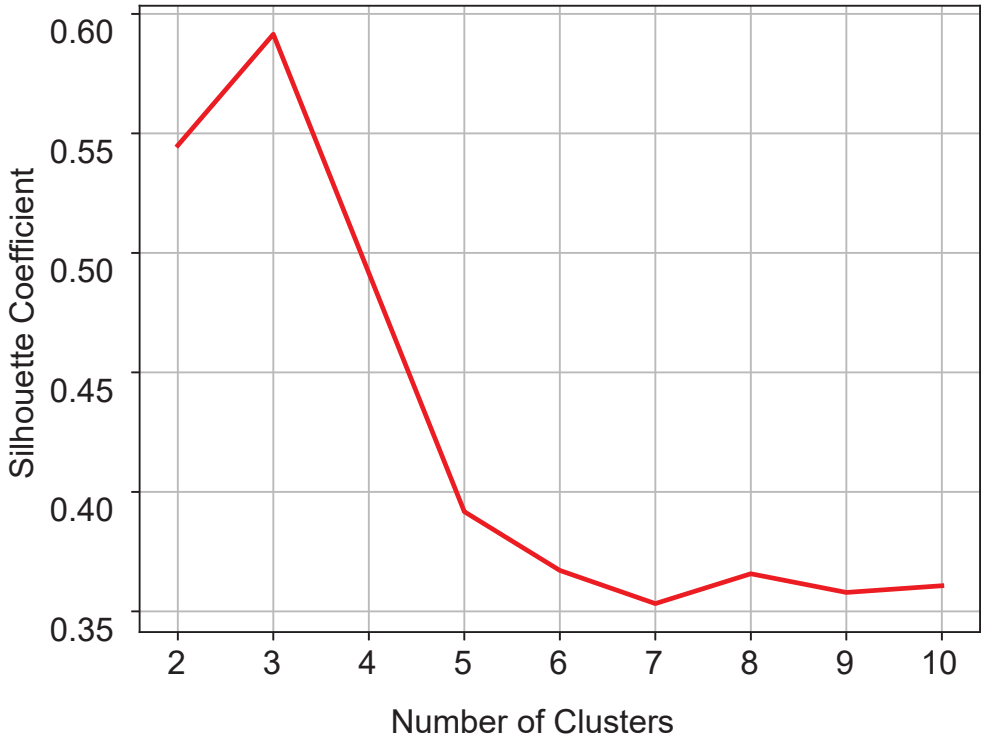

Figure 2.12: Silhouette scores.

As it is possible to notice, the K value on the elbow is the one with higher silhouette coefficients, but also K = 2 has a good silhouette value, thus can be considered a good candidate.

2.9 The Reinforcement Learning

Among all the different ML techniques, RL, at present, is one of the most attractive, with several application fields in which it was implemented [49]. In particular, the modelization of a RL policy is depicted in Figure 2.13.

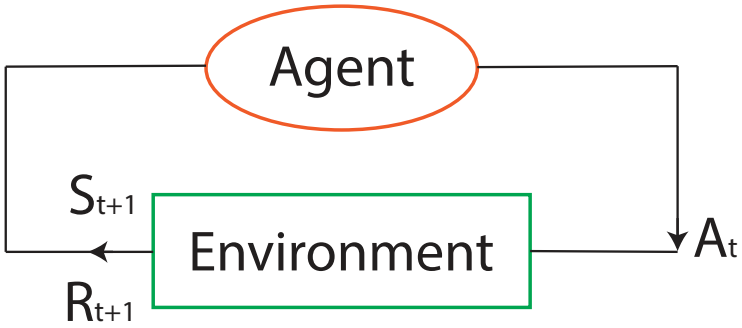

Figure 2.13: Reinforcement Learning (RL) components.

The environment (i.e. the system) is modeled through the definition of suitable states, being the states set named S. The RL policy foresees to train an Agent, to make it learn how to properly act on the environment. In particular, a set of actions A must be defined, being each action $A_i \in A$ a possible choice to allow the transition between states. In particular, at the time interval t, the chosen action A_t enact the transition between the initial state S_t and the final state S_{t+1}. The training phase of the algorithm is carried out by using a trial and error methodology, where the goodness of each chosen action is evaluated through the definition of a reward function, r. Summarizing, each RL technique foresees the definition of a suitable Markov Decision Process (MDP) ($MDP = \{S, A, P, r\}$), where P is the probability function. At each training step t the agent chooses an action, causing the transition between the state S_t and S_{t+1}. Actually, a reward R_{t+1} is produced (describing the validity of the performed action A_t) by evaluating the reward function, i.e. $R_{t+1} = r(t+1)$. Specifically, the aim is to fill a table where at each pair $\{(S_i, A_i) \quad | \quad S_i \in S, \quad A_i \in A\}$ is associated a suitable rate, evaluating the goodness of the action A_i given the state S_i. Afterwards, during execution, the so-called best policy π is exploited, by using (at each state S_t) the best evaluated action A_t. Usually, the function $V^{\pi}(s) = E_{\pi}\{R_t|S_t = s\}$ or the function $Q^{\pi}(s, a) = E_{\pi}\{R_t|S_t = s, A_t = a\}$ can be employed to evaluate the cumulative rewards associated with the specific best policy π, named respectively value and action-value function. In the formulas above, when performing the policy π, E_{π} represents the expected return value. Actually, the evaluation of either the $V^{\pi}(s)$ or $Q^{\pi}(s, a)$ functions,

can be carried out by using several different approaches. In the following chapters, when using RL, the Q-value associated to a specific pair (S_t, A_t), is calculated using Eq. (2.9).

$$Q(S_t, A_t) = Q(S_t, A_t) + \\ + \alpha * [R + \gamma * Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t)] \quad (2.9)$$

Specifically, Q-values are calculated by using $S_t, A_t, R, S_{t+1}, A_{t+1}$, namely State Action Reward State Action (SARSA) algorithm. Actually, during training, the Q-values are updated, while the best ones are used when exploiting π. In Eq. (2.9) the first step is to calculate the so-called *target*, $T = R + \gamma * Q(S_{t+1}, A_{t+1})$. In particular $\gamma \in \{0, \ldots, 1\}$ is chosen to suitably weight the importance given to the expected Q-value. Secondly, the learning rate α is used to properly weight the error between T and the previous Q-value. In particular, the higher α the higher the possibility to modify the Q-values. Usually, α and γ are set after a careful analysis of the system and a trial and error phase. Finally, during training, an $\epsilon - greedy$ algorithm (Eq. (2.10)) is used to evaluate a trade-off between exploration and exploitation.

$$\begin{cases} A_{t+1} = rand() \quad if \quad n \geq \epsilon \\ A_{t+1} = max(Q(S_t)) \quad if \quad n < \epsilon \end{cases} \quad (2.10)$$

In particular, exploration foresees to test a randomly chosen action, while exploitation foresees to use the actual best action, to assure that all actions are going to be evaluated during training. Referring to Eq. (2.10), $n \in [0, 1]$ represents a real random number, and ϵ can be suitably modified to achieve the desired trade-off between exploration and exploitation.

2.10 Evaluation Metrics for Classification problems

In the following chapters, supervised classification problems are considered. For this reason, the scope of this section is to introduce the evaluation metrics for such

problems and to briefly discuss a typical training activity. Usually, the whole dataset comprising inputs and (only for supervised ML techniques also the outputs) is splitted into training data and validation ones. Typically, $70-80\%$ of the whole dataset is used for training, while the other part to validate the model. During validation, the gold considered metrics is the accuracy, defined as per Eq. (2.11).

$$accuracy(\%) = \frac{y_t}{y_t + y_f} * 100. \tag{2.11}$$

In substance, accuracy represents the percentage of the correctly labeled data (y_t, where t stands for *true*) out of the total number of examples($y_t + y_f$, where f stands for *false*).

The accuracy analysis has one important limitation: it does not allow to analyze the inter-class classification errors. In particular, usually it is needed to understand when the algorithm makes classification mistakes. To this purpose, one solution could be the analysis of the confusion matrix, being Figure 2.14 one example, taken on the digits dataset.

In the main diagonal, it is possible to appreciate the correctly labeled data (the predicted label is equal to the actual one). In the other cells, it is possible to understand where mistakes are made. For example 3 "8" have been confused with "9". This analysis is particularly useful when dealing with an unbalanced dataset. Indeed, usually the dataset has to be balanced (i.e. each class has the same amount of examples). Practically, when dealing with real problems, sometimes it is impossible to collect a balanced dataset. In such situation could happen that the classification of the class with a lower number of examples is worst that the other classes. In this context, precision (Eq. (2.13)) and recall (Eq. (2.12)) can be exploited.

$$Recall_i(\%) = \frac{TP_i}{TP_i + FN_i} * 100. \tag{2.12}$$

$$Precision_i(\%) = \frac{TP_i}{TP_i + FP_i} * 100, \tag{2.13}$$

Where:

TP_i Represents the amount of True Positives (TP) in the class i, i.e. the number of examples belonging to class i, correctly labeled as i;

FP_i Represents the amount of False Positives (FP) in the class i, i.e. the number of examples belonging to class $j \neq i$, wrongly labeled as i;

FN_i Represents the amount of False Negatives (FN) in the class i, i.e. the number of examples belonging to class i, wrongly labeled as $j \neq i$.

Both $Recall_i$ and $Precision_i$ allows to properly evaluate the classification performances in each class, and ideally they must be both close to the maximum (100%). $Recall_i$ is a sort of class accuracy, calculated with the respect of the total number of examples belonging to class i. Similarly, $Precision_i$ is a sort of accuracy on the classification of class i, calculated with the respect of the total number of samples labeled as class i.

2.11 Training Process: a classification NN

Usually, the training of the ML algorithms is quite intuitive for the most of the aforementioned algorithms. The training process could be particularly tricky when dealing with NN. For this reason, the purpose od this section is to understand how to perform a training activity, with a special reference to an image classification problem solved with a NN.

Given the NN components already introduced in Section 2.5.1, the programmer must correctly choose:

1. The network architecture (numebr of layers, dimensions of layers...);
2. The activation functions;
3. Loss Function;
4. The optimizer;
5. The Learning Rate.

In particular, usually the programmer perform a trial and error activity, while the network is optimized by looking the metrics already introduced in the previous Section.

Generally speaking, both the loss and accuracy curves are used to evaluate the goodness of a NN. The latter has a more practical meaning, as it is strictly related to the number of correctly labeled examples, as per Eq. (2.11). The loss function is strictly related to the specific chosen loss function. As an example, Figures 2.15 and 2.16 respectively represent the loss and the accuracy plots derived from the training of a simple NN used to classify the famous fashion mnist dataset.

The curves depicted above represent, in general, a good training activity. They present a decreasing loss trend both in validation and in training and increasing accuracy curves over the different training epochs. An epoch represents a single training step, where all the training examples are given to the NN. The curves increase or decrease very slightly, and they stabilize in a final loss and accuracy value. If the curves are still changing, maybe it is better to increase the number of epochs or the LR. The LR represents how much the optimizer can change the perceptron's weights in each epoch. Keeping the LR low can make the network poorly trained, while an high learning rate can prevent the network from achieving the minimum value of the loss function. In the latter case, it is also possible to overfit the model. The overfitting is a condition where the network has learned too much from the examples given to it, thus resulting in a poor generalization capability. Finally, it is worth observing that usually training perform slightly better than validation.

By using convolutional layers on the fashion mnist dataset, it is possible to obtain the loss and accuracy curves of Figures 2.17 e 2.18.

This curves represent a situation where the model overfitted. It is possible to notice this behavior by analysing both the loss and the accuracy. The training is going increasingly well with epochs, i.e. the loss decrease and the accuracy increases. Despite this, validation is going worse, underlining the incapacity of the model to generalize to new data. A possible technique, to avoid overfitting, is the so–called dropout. This strategy foresee to disable some unuseful neurons in the hidden layer during training. Indeed, overfitting usually takes place when using a too-powerful network for a simple

problem. The disabled neurons are changed at each epoch, and the values of the disabled neurons are adjusted. In substance, it is like training different networks at each epoch, and then averaging the results of each NN.

With the dropout it possible to obtain the curves in Figures 2.19 and 2.20), that are better than the ones without convolutional layers.

It has also been demonstrated that when dealing with images it is better to normalize the input vectors (i.e. dividing each pixel value by 255.0). As a last remark, it is important to underline that the so-called LR, which represents how quickly the network learns, is always a key parameter, whose choice is usually done after several experimental tests, with a trial-and-error approach. Weights were updated at each step proportionally to the LR and to the calculated error..

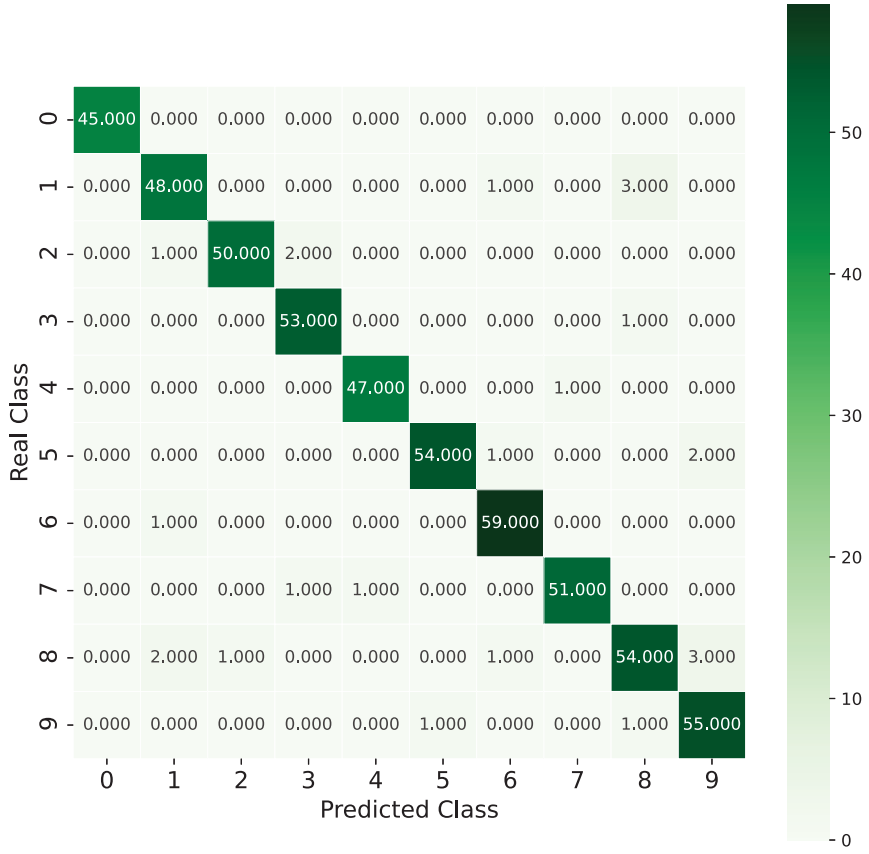

Figure 2.14: Confusion Matrix - Logistic Regression on "digits" dataset. Accuracy = 96%.

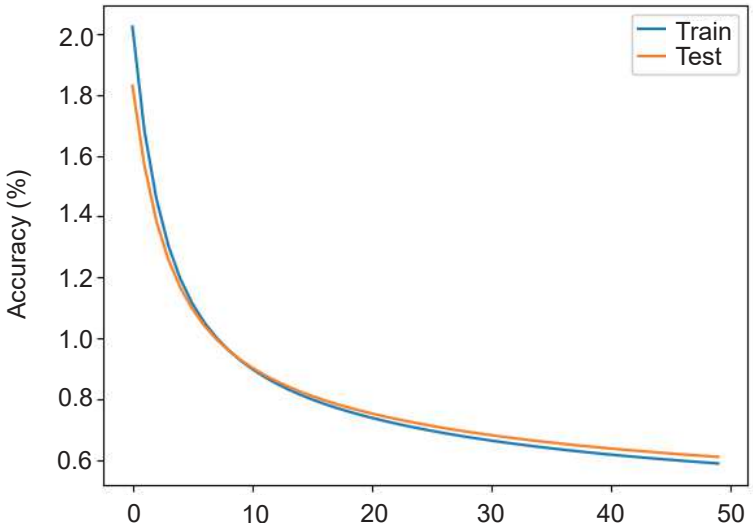

Figure 2.15: Loss (fashion mnist).

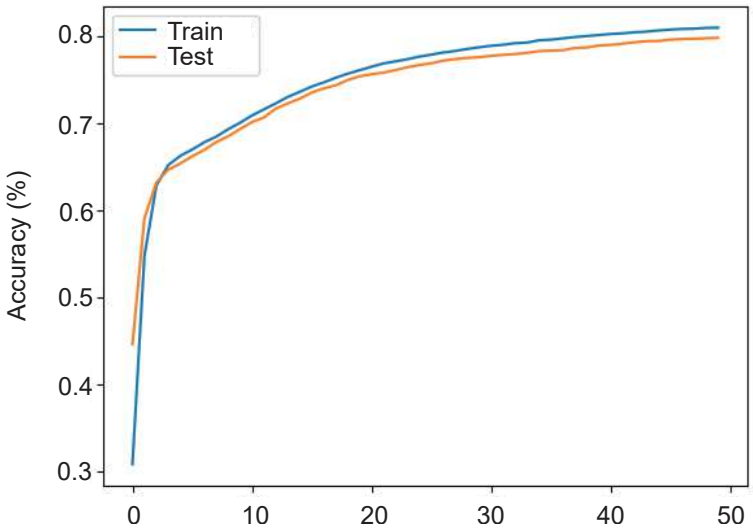

Figure 2.16: Accuracy (fashion mnist).

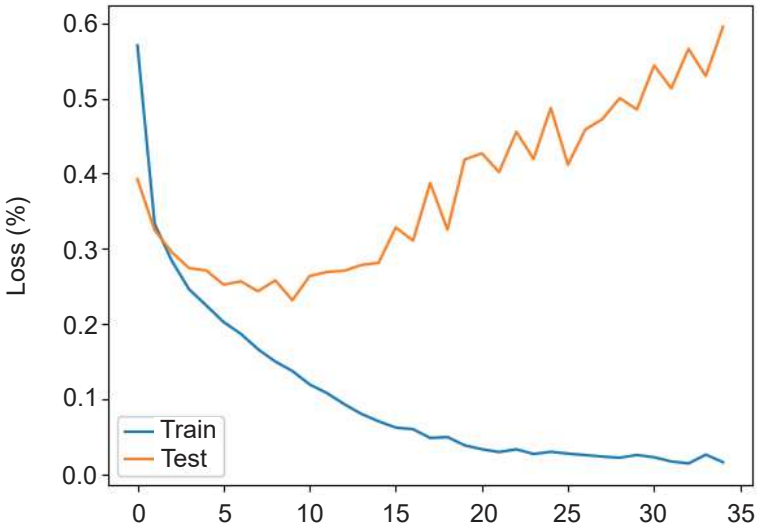

Figure 2.17: Loss (fashion mnist) - CNN.

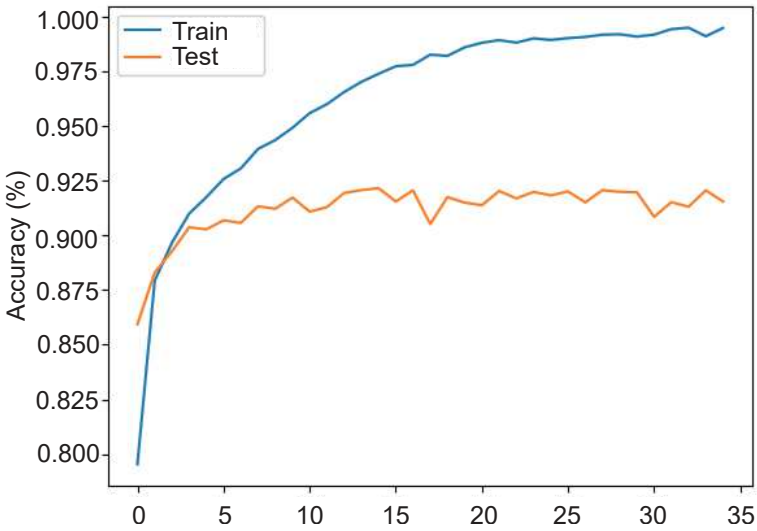

Figure 2.18: Accuracy (fashion mnist) - CNN.

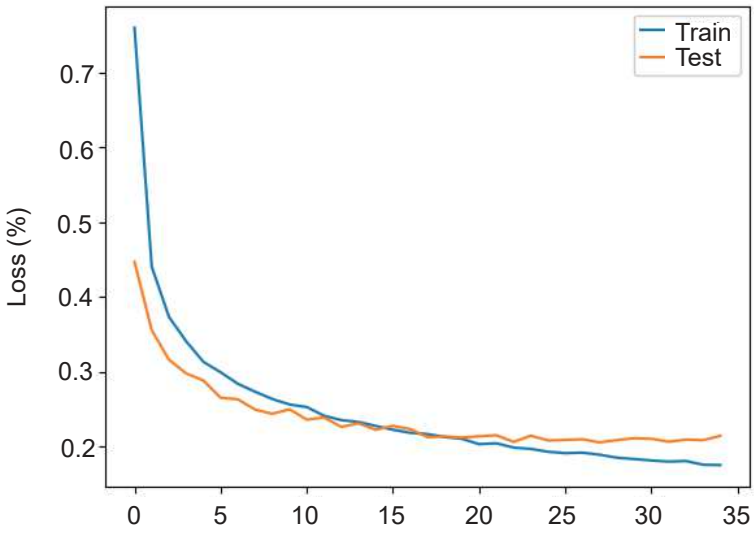

Figure 2.19: Loss (fashion mnist) - CNN and Dropout.

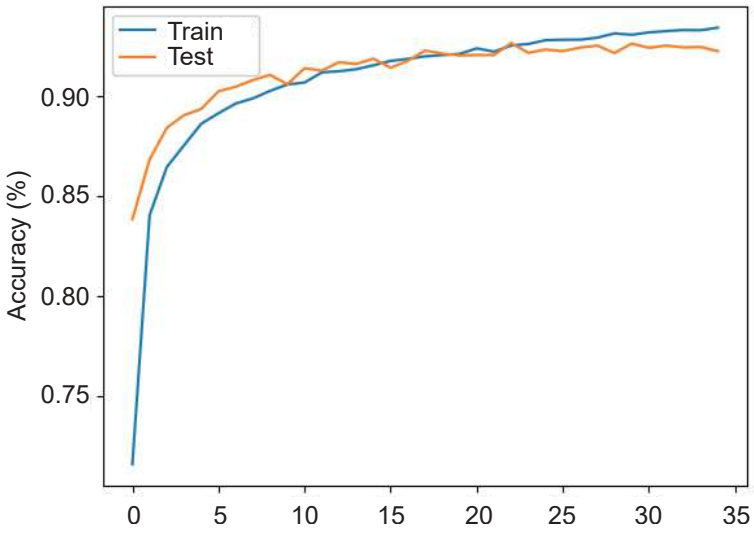

Figure 2.20: Accuracy (fashion mnist) - CNN and dropout.

Chapter 3

IoT Systems Communication Networks

This chapter aims at giving a general introduction to the communication networks that can be exploited in the industrial and I&M scenarios. The discussion will outline the need for an optimization of widespread communication networks coming from the general-purpose market, like Wi-Fi, LoRa or 5G.

3.1 Industrial Communication Networks

In the early days of industrial automation systems the need for data sharing among different parts of a machine soon brought to the design of dedicated communication systems, targeted at the industrial scenario, universally known as *fieldbuses* [50]. First installations of fieldbuses date back to the early 1970s, but the number of available solutions quickly diverged, to such an extent that it was referred to as a "fieldbus war" [51], where several manufacturers have proposed proprietary industrial communication protocols, often with similar but completely non-interoperable functionality. To overcome this fragmentation, lot of research energies were spent in standardization processes. Shelved the project to develop a unique communication system, in 1999 the first version of the IEC 61158 international standard was released, that comprised sev-

eral fieldbuses [52]. During the years, the IEC 61158 standard became a huge project collecting a lot of different fieldbuses, the majority of the total, for example Profibus, ControlNet, Interbus (only to cite a few). Significant limitations characterized these networks: low data rates, low number of connected nodes, as well as significantly reduced interoperability capabilities. Indeed, the integration of heterogeneous technologies and the sharing of data among different solutions were severely limited and internetworking capabilities were substantially absent [23].

With the subsequent proliferation of Ethernet technologies and the widespread availability of Internet connections, the automation world started to develop a new set of Ethernet–based systems, using the IEEE 802.1/802.3 specifications for the lowest communication layers. However, unless strict traffic and access controls are implemented, legacy Ethernet was unable to guarantee the required network latency, reliability and determinism. This intrinsic lack of real–time capabilities gave rise to the development of several dedicated (and proprietary) solutions, collectively referred to as RTE, or Industrial Ethernet, networks [53]. The IEC 61158 and IEC 61784 international standards gathered several of them, e.g. PROFINET, Ethernet/IP, Modbus/TCP, Ethernet POWERLINK, to name a few. Unfortunately, again the number of available RTEs rapidly increased, impairing interoperability, convergence, integration/implementation costs, substantially replicating the former fieldbus battle [54],[55].

Several shortcomings brought to this situation. Indeed, one of the major barriers to the realization of a "one fits all" solution was that different standardization bodies were involved in the design of a new RTE protocol, as well as *consortia* (e.g. Profibus, ODVA, etc.) has been formed to protect relevant market shares and brands. This resulted in a widespread adoption of RTE solutions in the last years, with a large industrial pervasiveness, but also in different approaches to obtain the desired performance. Indeed, irrespective to the market share, these consortia had no control over the standardization process of the underlying Ethernet (IEEE 802.1/802.3) standard, and often an RTE solution has been obtained introducing some protocol "hack" over the legacy Ethernet. Particularly, a well–accepted classification of different RTE systems follows the sketch in Figure 3.1, which identifies three different RTE classes with respect to different real–time performance [56].

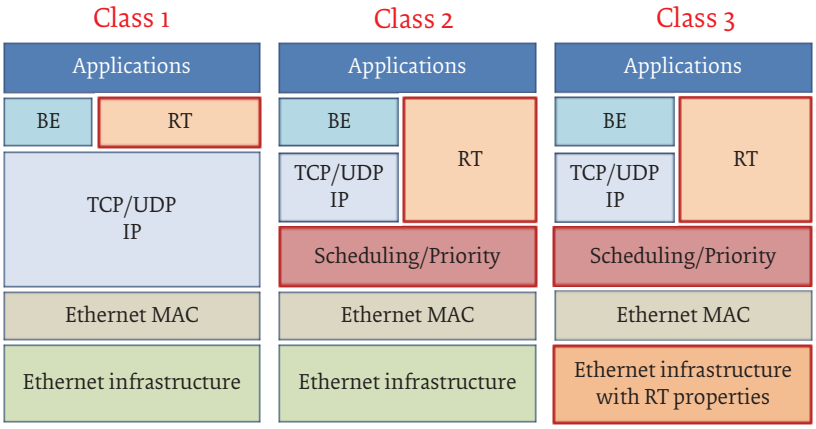

Figure 3.1: A widespread classification of RTE industrial networks.

For the aim to provide interoperation, several studies were made to connect different fieldbuses to each others or with different technologies using specific hardware or middleware protocol structure such as [57],[58],[59],[60]. The latter one is also an example of hybrid wired and wireless network, being a mixed network a key solution to develop smart measurement systems. Nowadays, how to adapt the widespread used fieldbus and RTE systems to the requirements of Industry 4.0 is still challenging. Several research activity has been made in this direction [61],[62]. Despite all, the complex technological panorama is so broad that the development of a pletora of adapters to interconnect different fieldbus and RTE system is practically infeasible. In this scenario, the development of new systems to use the CPS architecture and enact the Industry 4.0 revolution, is undoubtedly required [22].

As already said the IoM envisages a massive usage use of wireless communication strategies. In particular, wireless connectivity allows the possibility for typical industrial controllers to acquire information from sensors and send control signals to the actuators via a wireless communication system, building up the so–called Wireless Networked Control Systems (WNCS)[63],[64],[65]. As a matter of fact, two of the most important wireless solutions are represented by WirelessHART [66] and ISA100.11 [67], where the former is surely the most studied and complete one. In particular, three im-

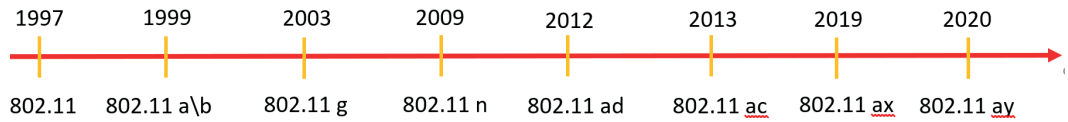

Figure 3.2: IEEE 802.11 standards.

portant articles deeply analyse a WirelessHART solution, providing high reliability also to the disturbances incurring on the controlled system. In particular, the formalization of the problem is carried out in [68], a distributed framework for the packet scheduling, $D^2 - PAS$, is presented in [69] and a fully distributed one, $FD - PAS$ is discussed in [70]. The main limit of this approach is the Time Division Multiple Access (TDMA)-based Data Link Layer (DLL) with a time slot of $10ms$, leading in an unacceptable fixed rating. In the last years, Wi–Fi reveled promising to be applied in factory automation as, compared with the aforementioned IEEE 802.15.4 solutions, it gives the possibility to cope with the timing requirements of the modern control systems and to perform a useful Rate Adaptation (RA) activity [71]. Indeed, for example authors of [72] underlined the necessity of a minimum control frequency of $1kHz$ for some specific application, not achievable by wirelessHART since it is characterized, as already said, by a time slot of at least $10ms$. How to adapt emergent wireless technologies, such as 5G and Wi–Fi, to the strict requirements of the factory automation is an open research field [73],[74],[75],[76], together with recent works concerning industrial LoRa networks [77].

3.2 Wi-Fi

The *IEEE 802.11* [78] is a set of standards, commonly known as WiFi, developed in order to be applied to the Wireless Local Area Network (WLAN). The physical layer of this standard is the perfect candidate to be applied in high-speed networks. The real time guarantees are not provided from the DLL of this standard, due to the random nature of the protocol used to regulate the channel access and the frame delivery [79]. Many versions of the standard have been developed, summarized in Figure 3.2.

The analysis of all the standards is out of the scope of this study, but a precise

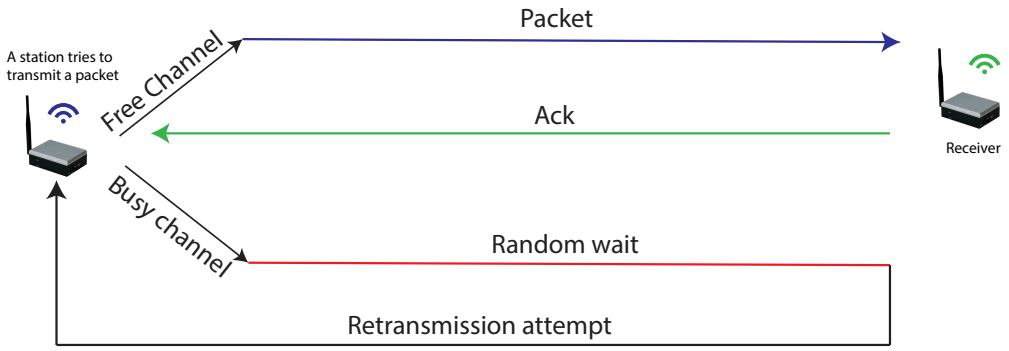

Figure 3.3: The DCF protocol.

analysis of the different protocols and layers is provided in [79]. In the following, the reasons why the Wi-Fi standard is not directly applicable for real-time applications are discussed. In particular, the communication time over the Wi-Fi network is not predictable, resulting in non deterministic behavior of the network. Indeed, the default channel access mechanism of Wi-Fi is essentially non-deterministic, mainly due to the unpredictable number of re-transmissions to be performed and the random delay between them. Two different concepts need to be discussed to understand why several retransmissions are needed. The channel access mechanism of Wi-Fi is graphically represented in Figure 3.3.

Figure 3.3 summarize the Distributed Coordination Function (DCF) protocol. The DCF policy is based on the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) algorithm with acknowledgement (ACK). Notice that the randomness depends not only from the random waiting time, but also from the number of retransmission attempts. Wi-Fi networks gives the possibility to use also a second channel access mechanism, namely Point Coordination Function (PCF). The PCF is a centralized algorithm where the central Access Point (AP) gives to a station the possibility to transmit. The waiting time before the transmission for each station is random also according to the PCF protocol.

The non-deterministic behavior of the network reflects in the impossibility to use these standards for industrial control applications without optimizing them.

3.2.1 Rate Adaptation Algorithms

The Multi Rate Support (MRS) feature of IEEE 802.11 can be deployed to dynamically change the transmission rate of a packet. Indeed, before each transmission, the adopted rate can be selected from a set of available rates.

Definition 3.2.1 (Set of available rates)**.** The set of the available rates, R, for the transmission of a packet is defined as follows.

$$R = \{r_1, r_2, ..., r_n\} \quad where \quad r_i > r_{i-1}, \quad i \in \{1, 2, ..., n\}.$$

□

Basically, the operation of a Rate Adapation Algorithm (RAA) consists in the selection of an ordered set of rates, called Retransmission Chain, to be used for the transmission of a packet.

Definition 3.2.2 (Retransmission chain)**.** Consider the transmission of a packet for which a maximum of N attempts can be carried out. The **Retransmission Chain**, RC, is defined as

$$RC = \{r^{(1)}, r^{(2)} ..., r^{(N)}\}, \quad r^{(i)} \in R.$$

meaning that the first transmission attempt is scheduled at rate $r^{(1)}$, the second at rate $r^{(2)}$, the N-th at rate $r^{(N)}$

□

3.2.2 ARF: Automatic Rate Fallback

Automatic Rate Fallback (ARF) [80] is one of the most popular rate adaptation strategies for Wi-Fi, designed for general purpose applications.

The behavior of ARF can be briefly described as follows. Consider a set R of available data rates. Starting from the generic rate r_i, a station implementing ARF moves to the lower rate r_{i-1} after K consecutive failed transmission attempts. Conversely, the rate is increased to r_{i+1} after N successful attempts. Also, if a failure occurs at the first transmission attempt after the rate has been increased, then the rate is immediately lowered down to r_{i-1} (this feature is known as *"Probing transmission"*). Typical values of the parameters K and N for the ARF algorithm are 2 and 10 respectively [81].

Unfortunately, ARF demonstrated to be rather ineffective for real-time industrial communication. However, its behavior allowed to derive two new algorithms able to reduce both the number of retransmissions and the delivery time of a packet, that revealed suitable for the industrial scenario. These are, namely, Static retransmission rate ARF (SARF) and Fast retransmission rate ARF (FARF) [82].

3.2.3 SARF: Static retransmission rate ARF

SARF [82] is a variant of ARF in which each retransmission attempt is carried out at rate r_1, i.e. the lowest rate in the set R. In practice, if the first transmission attempt fails, then for all the subsequent retransmissions we will have $r^{(i)} = r_1 \ \forall r^{(i)} \in RC$ with $i \in \{2, 3, ..., N\}$. However, in order to grant SARF with the same sensitivity to channel variations of ARF, a successful retransmission carried out at the lowest rate is not considered as an event that resets the consecutive failures counter. More precisely, only the original (first) transmission attempt is considered, and if K of these attempts fail then the rate will be lowered anyway, even if they were interleaved by one or more successful retransmissions. SARF ensures a very high success probability in packet transmission, since the lowest transmission rate uses the most robust modulation. This allows to limit the number of retransmissions, with the consequent reduction of the randomness. The pseudo-code of SARF is described by Algorithm 1.

Algorithm 1 SARF Algorithm

```
1: /* Initialization
2: ccf ← 0;                                   ▷ Number of consecutive transmission failures
3: ccs ← 0;                                   ▷ Number of consecutive transmission successes
4: R ← {r_1, r_2, ..., r_n};                  ▷ Available rates
5: i ← 1;                                     ▷ Select the lowest transmission rate in set R
6: tr ← r_1;                                  ▷ Set transmission rate to the lowest value
7: /* End Initialization
8: while true do
9:     if Request of a new transmission then
10:        if Previous attempt was failed then
11:            Perform transmission attempt at the lowest rate
12:        else
13:            Perform transmission attempt at the current rate
14:            if Success then
15:                ccf ← 0;                   ▷ Reset # of consecutive failures
16:                ccs + +;                   ▷ Increase counter of consecutive successes
17:                if ccs ≥ N then            ▷ If success threshold reached
18:                    if i < n then          ▷ If highest rate not reached
19:                        i + +;
20:                        t_r ← r_i;         ▷ Increase rate
21:                    end if
22:                end if
23:            else
24:                /* (If failure)
25:                ccs ← 0;                   ▷ Reset # of consecutive successes
26:                ccf + +;                   ▷ Increase consecutive failures
27:                if ccf ≥ K then            ▷ If failure threshold reached
28:                    if i > 1 then          ▷ If lowest rate not reached
29:                        i − −;
30:                        t_r ← r_i;         ▷ Decrease rate
31:                    end if
32:                end if
33:            end if
34:        end if
35:    end if
36: end while
```

3.2.4 FARF: Fast rate reduction ARF

FARF is an algorithm that represents a customization of ARF in which $K = 1$ and $r^{(i+1)} \leftarrow r_1$. In practice, just a single failed attempt is sufficient to set the new transmission rate to the lowest value in the set R, (i.e. r_1). Conversely, the increasing of the rate follows the same rules of ARF, and happens after N consecutive successful attempts. Analogously to SARF, $r^{(i)} = r_1 \ \forall r^{(i)} \in RC$ with $i \in \{2, 3, ..., N\}$. From a certain point of view, FARF is a more conservative algorithm than SARF, since it intrinsically assumes that a failure is an event that requires to maintain low rates for a certain number of subsequent packet transmissions. The pseudo–code for FARF is described by Algorithm 2.

Algorithm 2 FARF Algorithm

```
1:  /* Same code as SARF Algorithm (lines 3 to 7)
2:  while true do
3:      if Request of a new transmission then
4:          Perform transmission attempt
5:          if Success then
6:              /* Same code as SARF Algorithm (lines 16 to 22)
7:          else
8:              /* (If failure)
9:              ccs ← 0;                          ▷ Reset # of consecutive successes
10:             t_r ← r_1;                        ▷ Set transmission rate to the lowest value
11:         end if
12:     end if
13: end while
```

3.2.5 RSIN: Rate Selection for Industrial Networks

Rate Selection for Industrial Networks (RSIN) is a rate adaptation algorithm specifically conceived for the industrial environment, proposed in [83], and based on two fundamental assumptions. Specifically, i) a station wishing to send a frame (of a given length) has to be aware of the Signal to Noise Ratio (SNR) perceived by the receiver and, ii) the station has to know the relationship between SNR and Packet Error Rate (PER) for all the rates of the transmission set R. For each packet to be transmitted, let D be its deadline. Then, the outputs of RSIN are the number of retransmissions (N_{opt}), the retransmission chain (RC_{opt}) and the residual error rate probability for the transmission of that packet in a time less than D. More formally, RSIN can be described by a constrained optimization problem, as per Definition 3.2.3.

Definition 3.2.3 (RSIN constrained optimization problem)**.** For each packet transmission let be:

D the transmission deadline;

s the signal–to–noise ratio;

l the payload of the packet to transmit;

N_{max} the maximum number of retransmission attempts;

Let's define:

N the number of allowed retransmission attempts;

P_r the residual packet transmission error probability, given N retransmissions at rates $r^{(1)}....r^{(N)}$;

t_{trans} the transmission time.

Then, the constrained optimization problem is to find an optimal number of retransmissions attempts N_{opt}, and a retransmission chain $RC_{opt} = \{r_{opt}^{(1)}, r_{opt}^{(2)}..., r_{opt}^{(N_{opt})}\}$,

that minimize the residual packet transmission error probability, while ensuring that the deadline is not missed. This bring us to Eq. (3.1).

$$\begin{cases} \min\limits_{N \le N_{max}, r^{(i)}} P_r(l, s, N, RC) \\ \max\limits_{N \le N_{max}, r^{(i)}} t_{trans}(l, s, N, RC) \le D, \quad i = 1 \dots N \end{cases} \tag{3.1}$$

□

Algorithm 3 RSIN Algorithm

N ▷ number of retransmission attempts
/* ***Inputs*** */
N_{max} ▷ max number of retransmission attempts
$R = \{r_1, r_2, ..., r_n\}$ ▷ set of available rates
D ▷ Packet deadline
s ▷ SNR value
/* ***Definitions*** */
t_{trans} ▷ packet transmission time
$P_{r,opt}$ ▷ residual error probability for the optimal rate sequence
/* ***Definitions – Vectors of size*** N_{max} */
$\bar{P}_r$ ▷ $\bar{P}_r[i]$: residual error probability when $N = i, \quad i = 1 \dots N_{max}$
$\bar{t}_{trans}$ ▷ $\bar{t}_{trans}[i]$: transmission time when $N = i, \quad i = 1 \dots N_{max}$
$\bar{RC}_{low}$ ▷ Vector of rate chains.
/* The i-th element, $\bar{RC}_{low}[i]$, contains the retransmission chain of length $N = i$ that ensures the lowest transmission error probability*/
/* ***Outputs*** */
N_{opt} ▷ optimal number of retransmissions
$RC_{opt} = \{r_{opt}^{(1)}, r_{opt}^{(2)}, ..., r_{opt}^{(N_{opt})}\}$ ▷ optimal retransmission chain
for $i \leftarrow 1$ to N_{max} **do**
    Running Algorithm 4.
end for
/* Calculation of N_{opt} and RC_{opt} */
$N_{opt} \leftarrow 1$
$RC_{opt} \leftarrow \bar{RC}_{low}[1]$
$P_{r,opt} \leftarrow \bar{P}_r[1]$
for $i \leftarrow 2$ to N_{max} **do**
    if $\bar{P}_r[i] < P_{r,opt}$ **then**
        $N_{opt} \leftarrow i$
        $RC_{opt} \leftarrow \bar{RC}_{low}[i]$
        $P_{r,opt} \leftarrow \bar{P}_r[i]$
    else
        Do nothing
    end if
end for

Algorithm 4 Selection of the retransmission chain ensuring the lowest transmission error probability for each value of N

1: **repeat**
2: Generate a retransmission chain of size i, RC_i
3: calculate $t_{trans}(l, s \in S, N, RC_i)$
4: **if** $t_{trans}(l, s \in S, N, RC_i) \leq D$ **then**
5: **if** $P_r(l, s \in S, N, RC_i) < \bar{P}_r[i]$ **then**
6: $\bar{t}_{trans}[i] \leftarrow t_{trans}(l, s \in S, N, R_i)$
7: $\bar{P}_r[i] \leftarrow P_r(l, s \in S, N, R_i)$
8: $\bar{RC}_{low}[i] \leftarrow RC_i$
9: **else if** $P_r(l, s \in S, N, R_i) = \bar{P}_r[i]$ **then**
10: **if** $t_{trans}(l, s \in S, N, RC_i) < \bar{t}_{trans}[i]$ **then**
11: $\bar{t}_{trans}[i] \leftarrow t_{trans}(l, s \in S, N, RC_i)$
12: $\bar{P}_r[i] \leftarrow P_r(l, s \in S, N, R_i)$
13: $\bar{RC}_{low}[i] \leftarrow RC_i$
14: **else**
15: Do Nothing
16: **end if**
17: **else**
18: Do Nothing
19: **end if**
20: **else**
21: **Continue**
22: **end if**
23: **until** there are sequences of rates r_j, of length $i, \in R$, meeting condition 3.2

RSIN is described by Algorithms 3 and 4. One of the core sections of Algorithm 3 is represented by lines 18–20. At this stage, a vector of retransmission chains is built,

and each element stores the chain ensuring the lowest transmission error probability for that specific number of retransmission attempts, ranging from 1 to N_{max}. This is achieved with the invocation of Algorithm 4. Subsequently, lines 22–33 select the optimal retransmission chain among the N_{max} chains of the $\bar{RC}_{low}$ vector, as the one associated with the lowest transmission error probability. It is worth noting, in Algorithm 4, how the selection of the retransmission chain is handled in case more retransmission chains ensure the same transmission error probability. As can be seen (lines 9– 13), in this case Algorithm 4 selects the retransmission chain that has the lower transmission time. A further assumption has been adopted to limit the execution time of RSIN. Specifically, the construction of the array of retransmission chains by Algorithm 4, has been carried out under the condition

$$r^{(1)} \geq r^{(2)} \geq ... \geq r^{(N)} \tag{3.2}$$

that limits the retransmission chains to be considered.

Finally, in [71] a variant of RSIN, namely RSIN with Estimation (RSIN-E) has been introduced. RSIN-E has been devised to address the cases in which the SNR is not made available by the receivers. Thus, a learning algorithm has been designed to provide an estimation of the SNR, as better detailed in [71]. The learning algorithm is executed cyclically, with update period T_U. Clearly, the introduction of a further algorithm has the effect of increasing the execution time of RSIN-E, as will be investigated in the following. In particular, the lower the update period, the higher the impact on the execution time.

3.3 Time Sensitive Networking

The Industry 4.0 paradigm highlights the need for increasingly standardized and integrated networks [84]. In this context, TSN standards offer a viable solution, pointing to the development of a novel smart factory paradigm. The idea underlying the whole TSN project is to deeply modify the Ethernet standard at its roots (by the development of a new Ethernet Medium Access Control (MAC) layer and a new Eth-

ernet infrastructure), to introduce all those intrinsic mechanisms required to support a broad range of time-, mission- safety-critical applications. Indeed, on the contrary, all the available RTE networks build upon the legacy features of Ethernet, use protocol strategies (as a clever use of Virtual LAN prioritization) or even out-of-standard data–link layers to introduce real–time capabilities over a network support that is intrinsically non–real–time [85],[86].

Nevertheless, the first efforts in the stated direction have been pursued by the consumer electronics industry, and specifically for targeting the needs for deterministic Ethernet connections for professional audio and video streaming. This pushed towards introducing the needed modifications directly within the IEEE related standards. For this reason, in 2005 the Audio Video Bridging (AVB) Task Group (TG) was formed within the IEEE 802.1 standard committee. In parallel the AVnu Alliance has been formed, an associated group of manufacturers and vendors to support the compliance and marketing activities. The activities of the AVB TG allowed to strongly enhance the real–time capabilities of Ethernet with four new IEEE standards: 802.1AS-2011, 802.1Qat-2010, 802.1Qav-2009 and 802.1BA-2011. The new potentialities of Ethernet AVB were soon deemed suitable also for the industrial scenario [87]. For this reason, it was rapidly evident that the AVB name was not appropriate to cover all the potential use cases that the achievable performance attracted.

In 2012 AVB was renamed in TSN TG, a subgroup of IEEE 802.1 Working Group [88]. The suitability of these set of standards to different fields of application, has led to the definition of different *profiles*, that represent one of the most powerful characteristic of TSN. The IEEE 802.1 defines DLL protocols, as can be noticed from Table 3.1.

Table 3.1: IEEE 802.1 Contribution within IEEE 802.

ISO/OSI Layer	**IEEE 802 Standard**	
Data Link Layer	802.2 Logical Link Layer	
	802.1 Bridging	
	802.3 MAC	802.11 MAC
Physical	802.3 PHY	802.11 PHY

As it is possible to notice from Table 3.1, a network-specific MAC layer is located right under the 802.1 bridging layer. In this panorama, typically, two different LANs are considered: the IEEE 802.3 (Ethernet) and the IEEE 802.11 (Wi–Fi) one. TSN, traditionally, aims to enhance the performances of the IEEE 802.3 networks, but could also be applied to IEEE 802.11 networks, to reduce both delay and jitter [89]. The TSN standardization project focuses mainly on the IEEE 802.1Q (*IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks*) [90], with the development of several amendments to the standard. Indeed, time–sensitive traffic in different scenarios may have different QoS requirements, involving in the need of a set of configurable mechanism and protocols. Standards and amendments within the TSN project [91] are listed in Table 3.2.

Table 3.2: The TSN standardization project.

Standard	Description	Reference
IEEE 802.1AB	Station and Media Access Control Connectivity Discovery	[92]
IEEE 802.1AS	Timings & Syncronization	[93]
IEEE 802.1AX	Link Aggregation	[94]
IEEE 802.1CB	Frame Replication & Elimination	[95]
IEEE 802.1CS	Link Local Registration Protocol	[96]
Ongoing Projects		
IEEE P802.1CQ	Multicast and Local Address Assignment	[97]
IEEE P802.1DC	Quality of Service Provision by Network Systems	[98]
IEEE P802f	YANG Data Model for EtherTypes (amending IEEE 802-2014 [99])	[100]
IEEE P802.1ABcu	LLDP YANG Data Model (amending IEEE 802.1AB [92])	[101]
IEEE P802.1ABdh	Support for Multiframe PDUs (amending IEEE 802.1AB [92])	[102]
IEEE P802.1ASdm	Hot Standby (amending IEEE 802.1AS [93])	[103]
IEEE P802.1ASdn	YANG Data Model (amending IEEE 802.1AS [93])	[104]
IEEE P802.1CBcv	FRER YANG Data Model (amending IEEE 802.1CB [95])	[105]
IEEE P802.1CBdb	FRER Extended Stream Identification Funs (amending IEEE 802.1CB [95])	[106]

Amendments to the IEEE 802.1Q standard

Amendment	Description	Reference
802.1Qat	Stream Reservation Protocol (SRP)	[107]
802.1Qav	Credit based Shaper	[108]
802.1Qaz	Stream Resv. Pot.	[109]
802.1Qbu	Frame Preemption	[110]
802.1Qbv	Enhancements for Scheduled Traffic	[111]
802.1Qca	Path Control	[112]
802.1Qcc	TSN Configuration	[113]
802.1Qch	Cyclic Queuing	[114]
802.1Qci	Per–stream Filtering	[115]
802.1Qcp	Yang Data Model	[116]
802.1Qcr	Asynchronous Shaping	[117]
802.1Qcx	YANG Data Model for Connectivity Fault Management	[118]
Ongoing Projects		
P802.1Qcj	Automatic Attachment to Provider Backbone Bridging (PBB) services	[119]
P802.1Qcw	YANG Data Models	[120]
P802.1Qcz	Congestion Isolation	[121]
P802.1Qdd	Resource Allocation Protocol	[122]
P802.1Qdj	Configuration Enhancements for Time-Sensitive Networking	[123]

Amendments to the IEEE 802.3 standard

Amendment	Description	Reference
802.3br	Interspersing Express Traffic	[124]

In the Table the IEEE 802.3br amendment to the IEEE 802.3 standard is also reported, as the TSN preemption support requires a slight modification of the Ethernet standard. Moreover, in Table 3.2 are listed, among the others, several 802.1 ongoing projects, thus underlining that the TSN task group is still performing a ceaseless standardization activity. For this reason, Table 3.2 has not to be considered exhaustive and definitive. Moreover, it is worth observing that this thesis focuses on the most important standards for industrial measurement applications, and does not address all the aforementioned standards. This wide range of mechanisms and protocols offered by TSN, comprehensively aiming to reduce frame loss, synchronize stations among each others, provide bounded latency and high reliability [113], need to be precisely configured in each bridge of the considered network, to meet specific QoS requirements.

3.3.1 Time Sensitive Networking over Wi-Fi

The scheduling, bandwidth reservation, real–time behavior, Wi–Fi capabilities and other features of TSN, open up to interesting and advanced time–critical application where a constant flow of information, often coming from heterogeneous sensors, is of vital importance. An example is the scenario proposed by [125] where swarn of quadcopters are controlled to perform maneuvers at high speed. In this application, measurements from cameras and onboard sensors are used by a centralized control system to determine the references of each individual agent so that they can move in a coordinated way. Specifically, a system consisting of 8 cameras acquires the position and attitude of each vehicle with a frequency of 200Hz. The camera frames are sent via a User Datagram Protocol (UDP) stream to a central processing unit. Furthermore, each quadrotor is equipped with on-board sensors (accelerometer and gyroscope), the measurements are sent via an XBee–UDP bridge to the central processing unit. Here, they are processed, and each vehicle receives setpoints for coordinated motion via a PPM analog transceiver with a 50Hz refresh rate. Another communication channel is a low priority downlink for the purpose of data logging. The real–time requirements are evident since the failure to comply with a deadline or delays in the communication chain could lead to unexpected and catastrophic results. The use of different types of traffic, such as real–time and best effort, is also evident, with the separation achieved

through the use of physically separate communication channels. However, the communication architecture has some limitations. To maintain a sufficiently low latency and high bandwidth, the data flow from the cameras uses UDP which does not provide any QoS mechanism, exposing the system to potential packet losses. Using bridges to switch from UDP to other communication systems can represent an additional bottleneck. Both of these downsides are destined to become critical if the number of agents, and therefore the data flow, increases. In this context, some of TSN's features can bring benefits. For example bandwidth reservation and traffic scheduling can be used to prioritize video streams and cyclic data for the control system. The Frame Replication and Elimination for Reliability (FRER) can be used to increase the reliability of the communication. The use of these features allow to lower the network latency and jitter. Also the intrinsic clock synchronization required by TSN brings some advantages. Often in distributed autonomous systems GPS is used for clock synchronization in agents. TSN provides further improvements by providing a shared sub–microsecond time reference to the network's nodes, which can overcome GPS's existing constraints [126]. In addition, to the decrease in latency, communication times, and improve synchronization, a precise time–stamping of measured data can be used also to compensate for further delays introduced by the measurement, processing, and control chain.

As a matter of fact, the depicted scenario is a meaningful test case, where Wi-Fi network could be applied to cope with the requirements in terms of data rate. Despite this, the required real-time and deterministic behavior can be satisfied by the usage of TSN on top of the IEEE 802.11 DLL. Some works suggest the usage of hybrid wired/wireless networks, integrating ethernet TSN networks with both Wi–Fi [89] and 5G [127]. Actually, TSN over Wi–Fi networks are promising to adapt Wi–Fi to the stringent requirements of the industrial context. At present, the IEEE 802.11AS standard [93] specifically refers also to IEEE 802.11 LANs, providing a synchronization mechanism similar to the one designed for Ethernet networks, with the exception of some media–dependent activities specified in IEEE 802.1AS [93], Clause 12. In particular, how to communicate the timing messages between a Master Port and the attached Slave Port in the generated *spanning tree* is quite different with the respect to the full–duplex Point to Point links. In this case, in fact, the IEEE 802.11 [78]

Timing Measurement procedure is used to calculate the propagation time. The last version of the IEEE 802.11 standard allows also to use the Fine Timing Measurement mechanism [78]. The transposition of the other TSN features in WiFi is still an open research field.

3.4 LoRa and LoRaWAN

The aforementioned and widespread wireless technologies, such as Wi-Fi and 5G, are particularly suitable when there is the need for high data rates and Ultra Low Latency (ULL) behavior. As a matter of fact, some IoM applications need different requirements, such as long communication range and low battery consumption. In the introduction of Chapter 5 for example, a meaningful example test case is analyzed.

In these particular situations, other kind of wireless protocols can be analyzed. In particular Low Power Wide Area Networks (LPWANs) are gaining particular interest in the field, as they comprise several different wireless communication technologies, characterized by wide area coverage and low battery consumption. Both of these requirements are very important in the IoT scenario. These networks are also capable to guarantee robust communication even in harsh environments (as, for example, the industrial one), in consideration their particularly robust modulation schemes. Despite these important advantages, LPWANs are usually characterized by low data rates. Some of the LPWANs are license-free, thus operating typically in the Sub-GHz unlicensed spectrum regions, mostly exploiting in the EU the 868 MHz band.

One of the most used and studied technology is the LoRa Wide Area Network (LoRaWAN), that is built on top of the LoRa physical layer. In the next section, some important characteristics of both LoRa and LoRaWAN are discussed.

3.4.1 Main LoRa and LoRaWAN characteristics

As said before, LoRaWAN is one of the most used LPWANs operating in the unlicensed bands. The typical architecture of a LoRaWAN network is depicted in Figure 3.4.

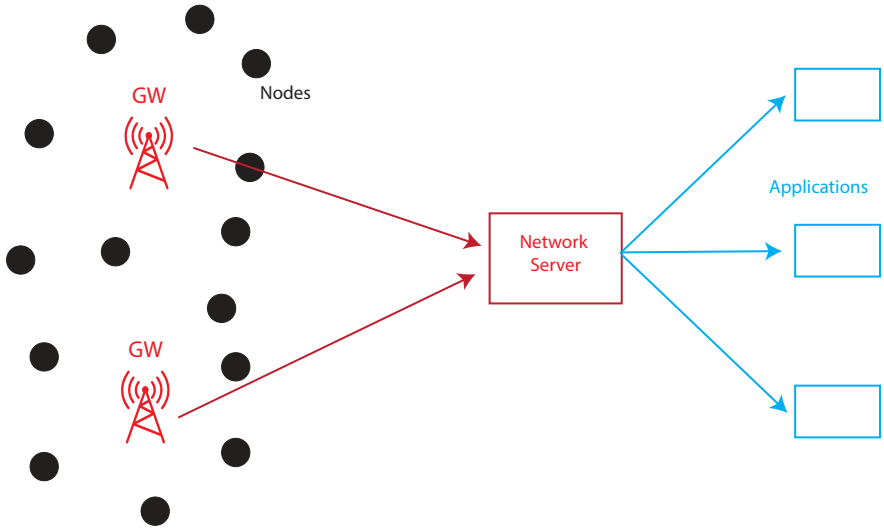

Figure 3.4: LoRaWAN Network Architecture

In particular, a LoRaWAN network relies on a star topology and comprises three types of devices: End Device (ED)s that serve as LoRa sensors, GateWay (GW)s, and a Network Server (NS). EDs are the data gatherers that send their information to one or more gateways in their range, using an ALOHA protocol to access the wireless medium. A GW receives a LoRa packet and forwards it to the NS that may either accept or discard it. In this way, most of the computational burden is handled by the NS, thus simplifying the complexity of the other devices (specifically EDs), with the consequent benefits, particularly in terms of power consumption. The EDs are grouped into three different classes: All(A), Beacon(B), Continuous(C). The three classes have a common transmit phase in which nodes can randomly access the network to transmit their data, whereas they differentiate on the subsequent phases. Class A EDs after transmission open one or two consecutive temporal listen windows, in which they can receive messages coming from the NS. Then class A EDs enter the sleep mode. Class B EDs extend such behavior possibly opening more listening windows at scheduled times. Finally, Class C EDs, after transmission enter the listening mode which is maintained until the end of the cycle. In practice, the sleep mode is never entered by such EDs.

Clearly, moving from Class A to Class C, the power consumption increases. It is worth observing that, the particular network architecture is particularly applicable to the IoM field, as the ED could be the sensing part of the network. In the following experiments, the optimization of the LoRa communication between the EDs and the GWs will be envisaged.

Talking about the LoRa physical layer, developed by Semtech, is essentially an enhanced version of the Chirp Spread Spectrum (CSS), being the BandWidth (BW) and the Spreading Factor (SF) its main parameters. In particular $2^{SF} chips$ form one symbol, and the relation between BW and SF is given by Eq. (3.3), representing the Symbol Rate (SR).

$$SR = \frac{BW}{2^{SF}} \tag{3.3}$$

Roughly speaking, the higher the SF, the lower the data rate. Choosing an high SF leads essentially to more robust (and more immune to noise), but slower communication. Furthermore, in EU the BW is 125kHz or 250kHz. Moreover, interleaving and whitening strategies are applied to increase interference robustness; packet header and payload are further coded by forward error correction, with Coding Rate (CR) in the range $CR \in [4/5, .., 4/8]$.

In a LoRaWAN network, the NS is in charge of choosing the LoRa network parameters, possibly applying adaptive strategies like the standard Adaptive Data Rate (ADR), which leverages on SF tuning and variable Transmission Power (TP), whose possible values are respectively listed in Eq. (3.4) and Eq. (3.5).

$$SF_{list} = [7, 8, 9, 10, 11, 12] \tag{3.4}$$

$$TP_{list} = [2, 5, 8, 11, 14] \quad (dBm) \tag{3.5}$$

3.4.2 Duty Cycle Limitations

LoRa operates on the Industrial Scientifical Medical (ISM) bands. Consequently, one of the main limitations on the application of LoRa for IoM is the DC limitation for

devices not adopting a listen-before-talk strategy [128]. Indeed, the channel access in the ISM bands is regulated by the reference standard EN 300 220-1, whose last version is V3.1.1 [129]. Actually, following the aforementioned reference standard the DC is set to 1%. For this reason, if the three mandatory channels are used (868.1, 868.3, 868.5), the duty cycle per each channels becomes $\frac{1\%}{3} = 0.33\%$, meaning that the inter-message delay needs to be suitably tuned, in order to fulfill regulations. Moreover, it is worth observing that, as stated by authors of [130], the higher the airtime, the higher both the probability of collisions and the clocks drift. The latter directly impacts on the possibility to efficiently schedule the communication, as stated by authors of [131], that also propose an interesting traffic schedule for LoRa. Moreover, the non-perfect synchronization of stations impacts on the modulation / demodulation activity of LoRa [132].

Actually, the objective now is to suitably formalize how it is possible to calculate the inter-message delay, and in turn the number of messages allowed to be sent in an hour. The first step consist the airtime calculation (i.e. the time needed to transmit a message from a node to the GW). The airtime can be calculated simply as the ratio between the number of sent symbols n_s and the SR, as per Eq. (3.6). Actually, the propagation time is negligible when considering industrial applications where distances are very low (some meters).

$$airtime = \frac{n_s}{SR} \quad (ms) \tag{3.6}$$

Where the number of symbols n_s, that is calculated in Eq. (3.7), is the sum of the payload and preamble ones.

$$n_s(symbols) = n_p + 4.25 + 8 + max\left(\lceil k \rceil \cdot (4 + CR), 0\right) \tag{3.7}$$

And:

$$k = \frac{8 \cdot PL - 4 \cdot SF + 8 + 16 \cdot CRC + 20 \cdot H}{4 \cdot (SF - 2 \cdot LDR)} \tag{3.8}$$

Where:

n_p is the number of symbols of the preamble (tunable by the user);

PL represents the data PayLoad in bytes;

CRC is 1 if the Cyclic Redundancy Code field is enabled, otherwise 0;

H is 1 if the Explicit Header field is enabled, otherwise 0;

LDR is 1 if the Low Data Rate optimization is enabled, otherwise 0. It is worth observing that LDR can be enabled only if BW = 125 kHz and $SF \geq 11$;

Finally, remember that the SR can be calculated as per Eq. (3.3).

After the airtime has been calculated the intermessage delay according to the considered DC can be simply derived from Eq. (3.9) which, in turn, determines the number of possible sent messages in an hour.

$$delay = airtime \cdot (1 - DC(\%)) \quad (ms) \tag{3.9}$$

An example of the aforementioned timings, calculated for a specific set of parameters, but for all the possible SF, is presented in Table 3.3.

Table 3.3: Data rate, airtime, inter-message delay, and the number of messages per hour for each Spreading Factor. Settings: BW = 125.0, CR = 1.0, Payload (bytes) = 5.0, DC(%) = 1.0.

SF	**DR**(kbit/s)	**Airtime**(ms)	**Inter-message Delay** (s)	**#msgs/h**
7	5.469	36.096	3.574	1007.415
8	3.125	72.192	7.147	503.707
9	1.758	123.904	12.266	293.482
10	0.977	247.808	24.533	146.741
11	0.537	495.616	49.066	73.371
12	0.293	827.392	81.912	43.950

3.4.3 The ADR strategy

As stated before, the NS is in charge of changing the transmission parameters of LoRa radios, in particular the TP ans the SF. The LoRaWAN specifications define the so-called ADR parameter adaptation strategy that is responsible for managing the SF and TP of EDs.

ADR bases its behavior on an estimation of the so-called uplink signal-to-noise ratio margin (Eq. (3.10)).

$$SNR_M = SNR_{Meas} - SNR_{DR} - M \tag{3.10}$$

Where SNR_{Meas} is the maximum measured SNR over the last 20 received uplink frames, SNR_{DR} is the required SNR for successfully decoding incoming frames at the desired data rate DR and usually $M = 10dB$. Subsequently, the discrete parameter $NStep = int(SNR_M/3)$, being int the integer part, is computed. It is important to observe that, on the ED side, TP and SF are reduced as much as possible, and the ED exploits acknowledgment from the backend to regain connectivity once it is lost. ADR_ACK_LIMIT (by default equal to 64) and ADR_ACK_DELAY (by default equal to 32) regulate ADR on the ED. In particular, a counter (namely ADR_ACK_CNT) is incremented after each uplink message is transmitted. A request for an ADR_ACK is sent from the ED when the ADR_ACK_CNT reaches the ADR_ACK_LIMIT without receiving any downlink message. The ED then waits for the requested downlink message in the subsequent ADR_ACK_DELAY receiving opportunities. If a downlink message is not received after ADR_ACK_LIMIT + ADR_ACK_DELAY uplink frames, the TP is increased directly from the ED, up to the maximum value. If this is still not sufficient, also the SF is increased, to leverage on the additional processing gain [133].

Chapter 4

Optimizing WiFi for IoM

In the previous sections, an introduction on IoT-based measurement systems, or IoM, has been given, together with an analysis of its enabling technologies. One of the main issues related to IoM is related to the communication protocols, especially wireless ones. Wi-Fi, already introduced in Section 3.2, is surely one of the most used wireless LAN. In this section, optimization strategies for Wi-Fi networks are investigated, by using both experimental and simulation assessments.

4.1 Parameter Adaptation Strategies Execution Times

In Section 3.2.1, RAAs have been introduced. The following experiment's aim is to properly evaluate the execution times of the most used RAAs. Indeed, although if several RAAs have been proposed and analyzed during the years, up to my knowledge no one investigated the impact of the execution times of such algorithms. Obviously this analysis is needed, as the benefits coming from the RA strategy could be vanished if its execution time is too high. For these reasons, the main aim of this Section is to carefully analyze the behavior of the RA algorithms, by also taking into account the execution times of the implemented techniques.

4.1.1 Experimental Set–up

The RAAs described in Section 3.2.1, with the exclusion of ARF, have been practically implemented on a Personal Computer (PC) (Dell Optiplex 960) equipped with an Ubuntu Long Term Support Linux distribution, based on Linux kernel version 4.1.0-040100-lowlatency. The PC is equipped with two commercial off–the–shelf Wireless Network Interface Cards (WNIC)s, namely TP-LINK model TL-WN851ND, to implement an IEEE 802.11n network, as described in Figure 4.1.

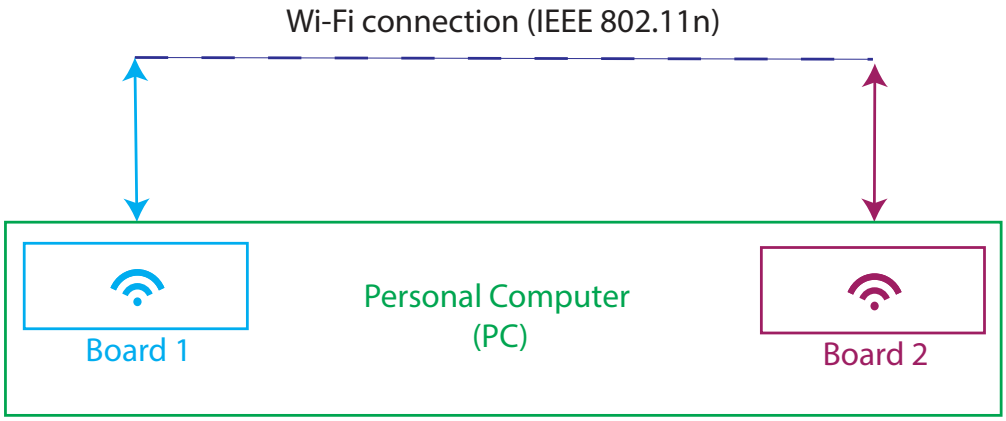

Figure 4.1: Experimental Set–up Representation.

The used WNICs are based on the AR9287 chip, which adopts the open–source `ath9k` driver. This choice, along with adequate modifications to the `mac80211` module, following the technique outlined in [134], allowed to implement and test the designed rate adaptation algorithms.

Clearly, the use of a PC with two WNICs does not reflect a real industrial network set–up at all. However, as it will be better detailed in the following, the first set of experiments carried out in this Section, do not need to be performed on an actual industrial network. Indeed, the performed analysis is aimed at evaluating the execution times of the RAAs and their impact on the MAC layer execution times. Such times are related to the algorithms implementation and do not depend on the network configuration. Moreover, having the two WNICs installed on the same PC revealed definitely

advantageous, because it allowed to exploit a unique clock source for the experiments, ensuring a high accuracy of the results, since there was no need to synchronize the two boards. Finally, by using a PC it has been possible to adequately modify the MAC layer protocol stack, as well as to enable/disable and tune the rate adaptation algorithms during the execution of the tests. However, in real I&M applications different devices may be used. For example, factory automation applications make use of Programmable Logic Controller (PLC), sensors/actuators and, possibly, Soft PLCs. All these devices, typically, do not grant full access to the MAC layer functions and, hence, could have not been used for the purposes of this Section. Nevertheless, and more importantly, rate adaptation algorithms do exist and are (or may be) used by such kind of devices. In this respect, I believe the results of the practical experiments carried out, which will be discussed in the next Section, represent a meaningful indication of the impact of RAAs on the performance of industrial devices.

For the specific case of the RSIN algorithm, two different implementations have been considered. In the first one, the constrained optimization problem (Eq. (3.1)) is run before the transmission of each packet.

In the second implementation, referred to as RSIN Light (RSIN-L) in the following, a set including all the optimal retransmission chains, for each possible SNR value, is determined at an early initialization phase, and is stored within a look–up table. Thus, the selection of the retransmission chain, that happens just before a single packet transmission, simply involves an access to that table. This strategy clearly reduces the execution time of the algorithm. However, since the constrained optimization problem is not solved dynamically, RSIN-L implicitly assumes that i) the relationship between SNR and PER does not vary over time, ii) the deadline is the same for all the transmitted packets, iii) the packets have fixed length. The actual assessment of such conditions depends on the communication channel status as well as on the specific application.

The RAAs have been implemented within the context of the Linux `mac80211` framework. In agreement with this approach, I developed the set of required functions for each algorithm. A test application has been subsequently designed in which the data exchange described in Figure 4.2 was triggered periodically, thus resembling a typical industrial polling sequence.

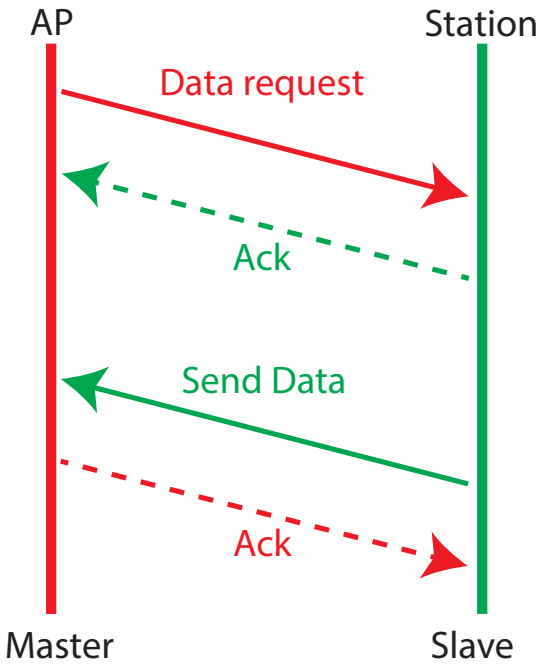

Figure 4.2: Data transaction between the two WNICs.

A Master–Slave relationship has been set–up where one board (the Master), configured as AP, sends a request packet to the other one, configured as Station (the Slave), that responds with a data packet. Each transaction ends when the Master receives the packet with the requested data. The complete list of the settings used for the experiments is summarized in Table 4.1. They have been selected to comply with the most common features/requirements of Wi–Fi based real–time industrial networks and their relevant rate adaptation algorithms, as described in [135], [27], [136] and [50].

In order to measure the RAAs execution times, I introduced adequate timestamp points within the relevant parts of the protocol stack code. A representation of the timings related to the transmission of a packet is provided in Figure 4.3.

As can be seen, the packet is initialized at time t_1, and delivered to the MAC layer at time t_2. The execution of the RAA takes place in the interval $t_3 - t_4$. The packet is eventually delivered to the physical layer at time t_5. Notably, the reference clock

Table 4.1: Settings of the experimental testbed.

Parameter	Value
Period of Transactions	10 (ms)
Number of transactions	100000
Packet Length	50 bytes
K_{SARF}	2
$N_{SARF} = N_{FARF}$	10
Update Period RSIN-E, T_U	10 (ms)
RSIN and RSIN-E Deadline	2 (ms)
Center Frequency	2660 (MHz)
Channel model	IEEE 802.11n
Channel bandwidth	40 (MHz)
STBC	Enabled
LDPC	Disabled
Guard Interval	Normal
N of spatial streams	2
Available Data rates [*Mbps*]	13.5, 27, 40.5, 54, 81, 108, 121.5, 135

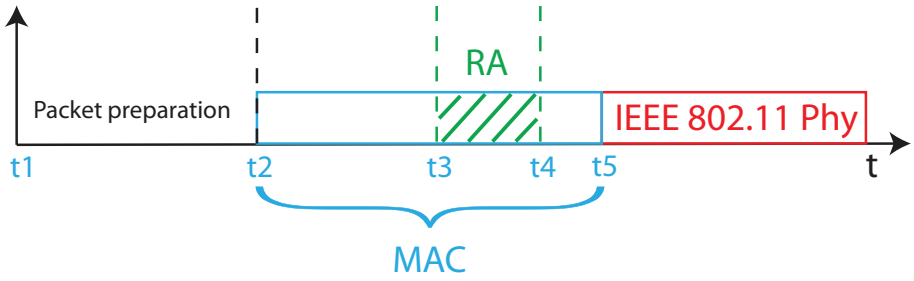

Figure 4.3: Timings related to the transmission of a packet.

shared by the two WNICs allowed to precisely timestamp the events concerned with code execution.

The correctness of the RAAs implementation can be verified by curves like the ones in Figures 4.4 and 4.5, respectively for FARF and RSIN. The other ones are not presented for sake of brevity. Each Modulation Coding Scheme (MCS) value, represented in the y-axis of the figure, represents a particular modulation scheme of Wi-Fi. The higher the MCS, the higher the rate.

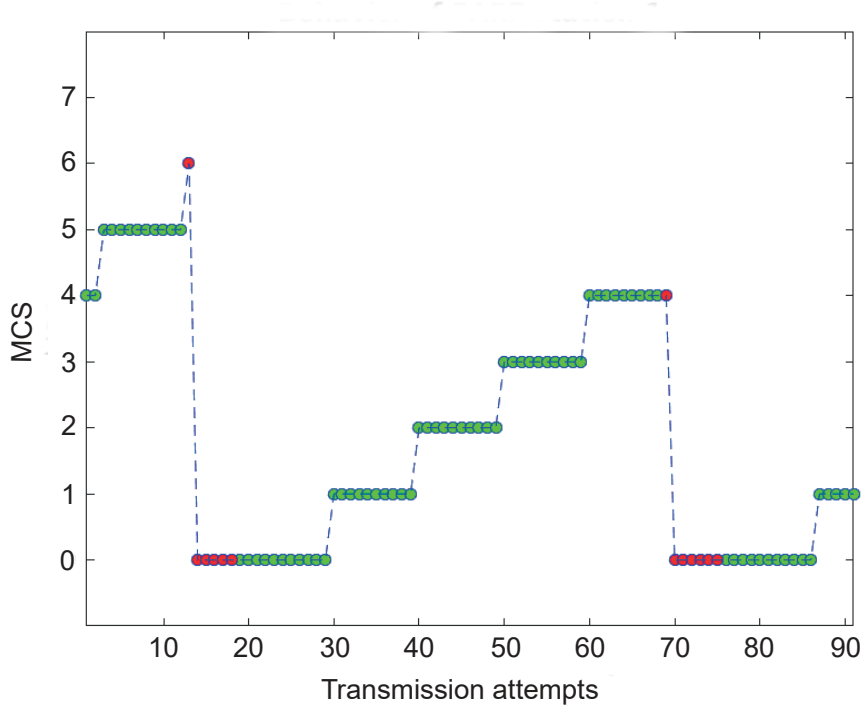

Figure 4.4: Rate Adaptation behavior of the FARF algorithm.

4.1.2 Execution Times of Rate Adaptation Algorithms

The execution time of each RA algorithm has been measured placing timestamps at the instants t_3 and t_4 in Figure 4.2. The obtained statistics are given in Table 4.2, whereas Figure 4.6 reports the Experimental Cumulative Distribution Function (ECDF).

As can be seen, the statistics also comprise Minstrel, a widespread general purpose algorithm [134]. Minstrel has not been designed for real-time industrial applications. However, it has been included here as an effective basis for comparison. From the presented results, it appears evident that SARF, FARF and RSIN-L have execution times of the same order of magnitude of Minstrel. Conversely, RSIN has the highest execution time, as an effect of the constrained optimization problem algorithm executed before each packet transmission. RSIN-E has an intermediate value, due to the

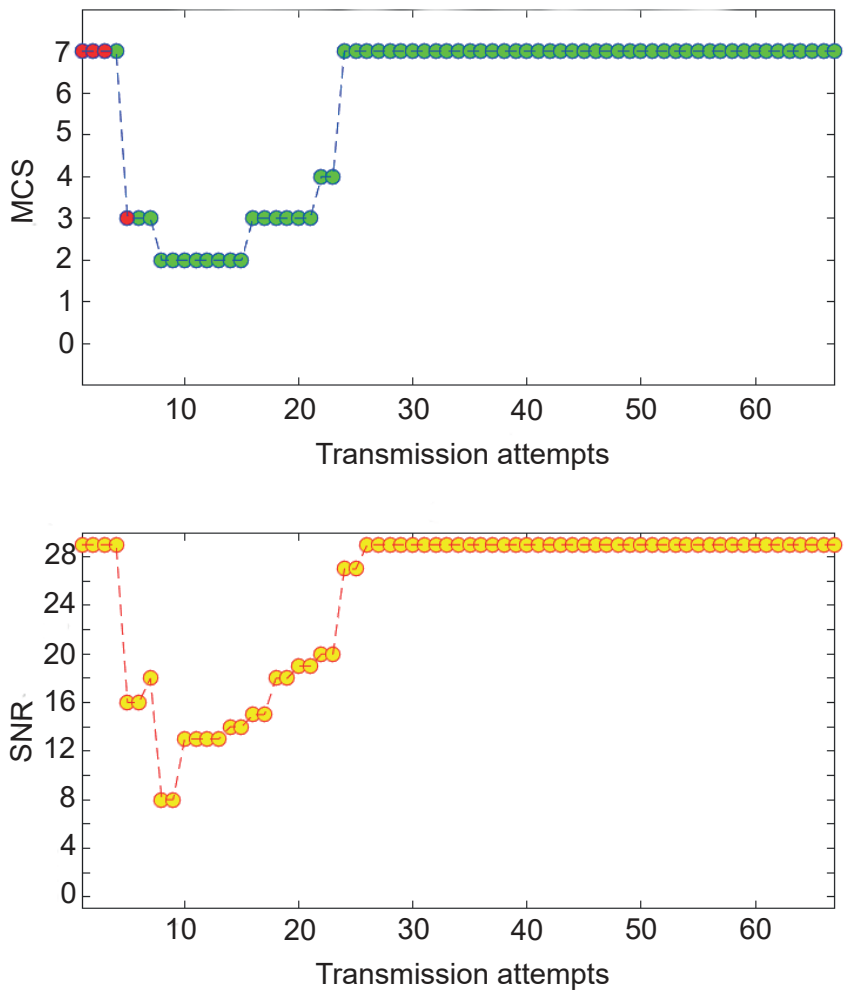

Figure 4.5: Rate Adaptation behavior of the RSIN algorithm.

choice of using a look up table in the experiments for such algorithm, analogously to RSIN-L. A higher value is expected in case the constrained optimization problem is executed before each packet transmission. Also, the mean execution time of RSIN-E can be reduced by increasing the update time of the learning algorithm, T_U, which in this experiment was set to a rather low value (10 ms, as reported in Table 4.1).

Table 4.2: Execution time statistics of the rate adaptation algorithms.

Algorithm	**Mean**	**Std. deviation**
SARF	1.56895 (µs)	0.993964 (µs)
FARF	1.80424 (µs)	0.999401 (µs)
RSIN-L	1.95316 (µs)	1.24508 (µs)
RSIN	22.8111 (µs)	2.9518 (µs)
RSIN-E	5.5637 (µs)	3.98806 (µs)
Minstrel	1.29763 (µs)	1.79429 (µs)

In industrial applications, in order to assess real–time performance, often worst case communication times have to be taken into consideration. For this reason, I calculated the maximum execution times of the algorithms, that are reported in Table 4.3. As can be seen SARF, FARF and RSIN-L perform better than Minstrel, whereas both RSIN and RSIN-E show higher times, in agreement with the previous analysis.

Table 4.3: Maximum values of the execution time of the selected RAAs.

Algorithm	**Max Value**
SARF	4.81206 (µs)
FARF	4.33086 (µs)
RSIN-L	6.13538 (µs)
RSIN	31.7115 (µs)
RSIN-E	17.2933 (µs)
Minstrel	7.71133 (µs)

4.1.3 Impact of the Rate Adaptation Algorithms on the MAC layer Execution Time

In a further session of experiments, I determined the execution times of the MAC protocol with the rate adaptation algorithms running, and evaluated the relevant impact of the algorithms themselves on such times. The *impact* is defined as the percentage of MAC layer execution time used to run the RA algorithm, and expressed as the ration between the algorithms execution time and the overall execution time of the

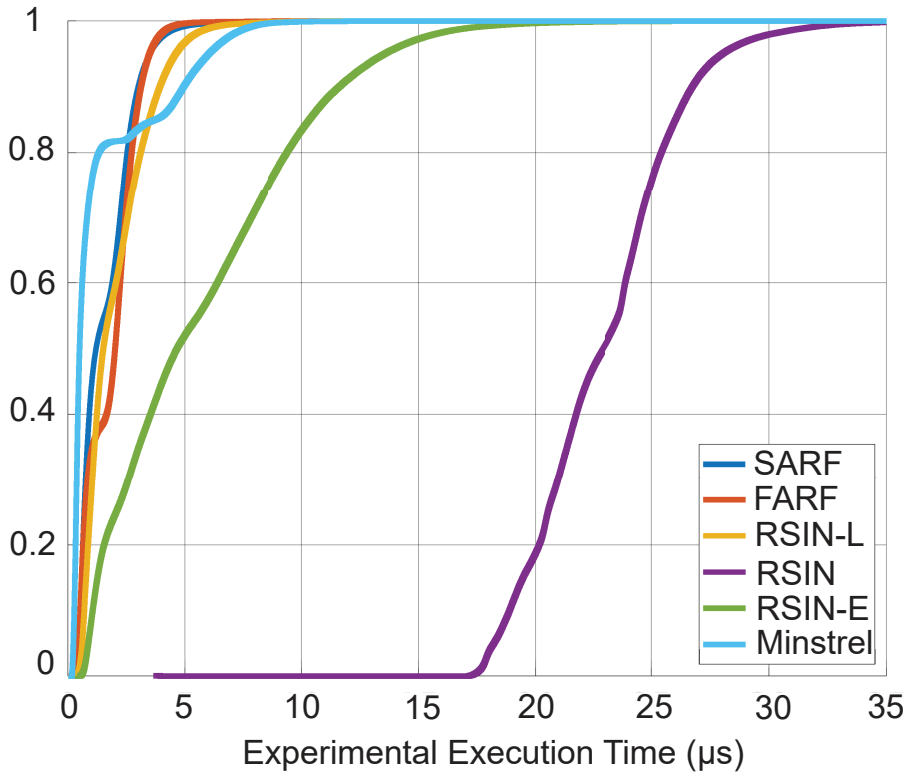

Figure 4.6: ECDFs for the comparison of the RAAs execution times.

MAC protocol stack. The MAC layer execution time has been measured, with refer to Figure 4.3 as the difference between the two timestamps t_2 and t_5. The statistics are reported in Table 4.4, whereas Figure 4.7 shows the ECDFs curves. The Table also reports the impact of RAAs.

As can be seen, the experimental results reveal that the impact of the rate adaptation algorithms on the IEEE 802.11 MAC layer execution times may be considerable, depending on the selected algorithm.

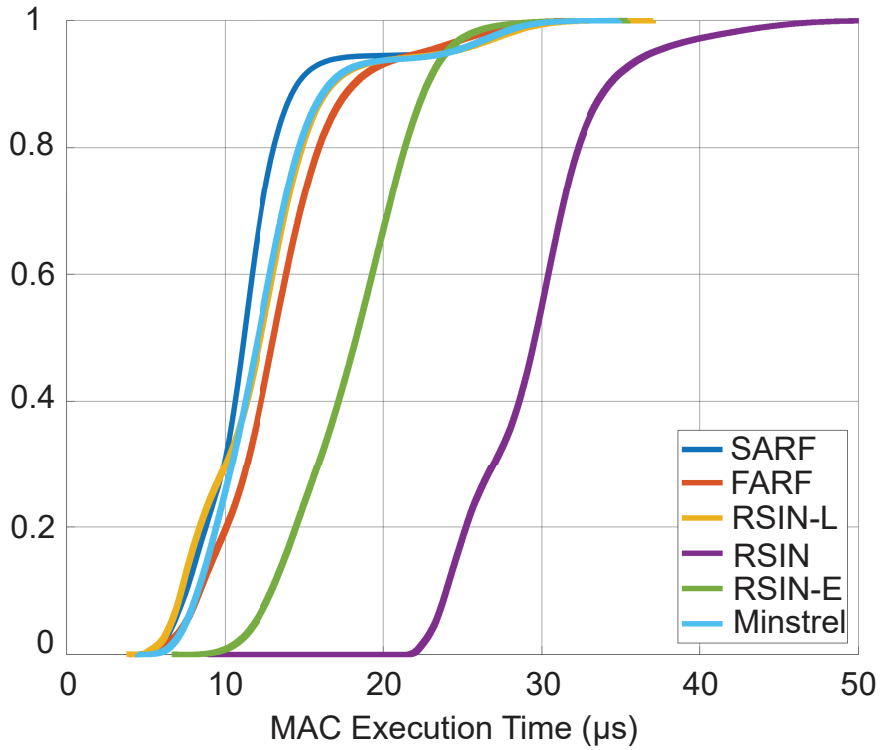

Figure 4.7: ECDFs representing the IEEE 802.11 MAC layer execution times.

Table 4.4: Impact of the RA techniques on the MAC layer execution time.

	MAC Execution Time		
Algorithm	**Mean**	**Std. deviation**	**Impact** [%]
SARF	11.7137 (µs)	6.26242 (µs)	13.39
FARF	13.4602 (µs)	6.81398 (µs)	13.40
RSIN-L	12.5359 (µs)	6.9672 (µs)	15.58
RSIN	29.4723 (µs)	7.00321 (µs)	77.40
RSIN-E	18.096 (µs)	6.90356 (µs)	30.75
Minstrel	12.7162 (µs)	6.91258 (µs)	10.20

4.1.4 Impact of the Rate Adaptation Algorithms on Communication

In order to achieve deeper insights about the various components of a data exchange cycle, I evaluated the time taken by the communication sequence of Figure 4.2, that in the following I refer to as Round Trip Time (RTT), and the consequent impact of the rate adaptation algorithms on it. Similarly to the previous section, the impact is evaluated as the percentage of RTT used to run the RA algorithm. In this experiment, I decided to use the same network configuration shown in Figure 4.1 that, as already observed, does not reflect that of a typical industrial network. Nonetheless, the obtained results are meaningful. Indeed, the proximity of the two WNICs ensures an efficient communication, characterized by the selection of high transmission rates by the RAAs and very limited packet retransmissions, as I assessed in a post processing analysis. Consequently, the communication times resulted definitely lower than those achievable with a (more realistic) distributed industrial network. Thus, the measured RTT represents a lower bound for such a performance index. Conversely, the measured impact is an upper bound, since the execution times of the rate adaptation algorithms do not depend on the network configuration.

The RTT has been calculated inserting two timestamps in the code, the first one placed at time t_1 (in Figure 4.3), and the second one, with refer to Figure 4.2, at the instant of data reception from the Slave. The resulting ECDFs are presented in Figure 4.8.

At a first glance, from the figure, it appears evident that the adoption of rate adaptation algorithms specifically designed for real-time communication algorithms results definitely advantageous, since they allow to achieve low values of round trip times, characterized by very limited variability, as also confirmed by the statistics reported in Table 4.5. This is due to the very high probability of success at the first transmission attempt they achieve (for example, $98,2\%$ for SARF and 99.6% for RSIN have been measured). In this way, the random backoff times between two subsequent transmission attempts were mostly avoided, with the consequent benefits on timeliness [83].

More importantly, Table 4.5 shows that the impact of the execution time of all the rate adaptation algorithms on RTT is very limited, in particular, also for algorithms

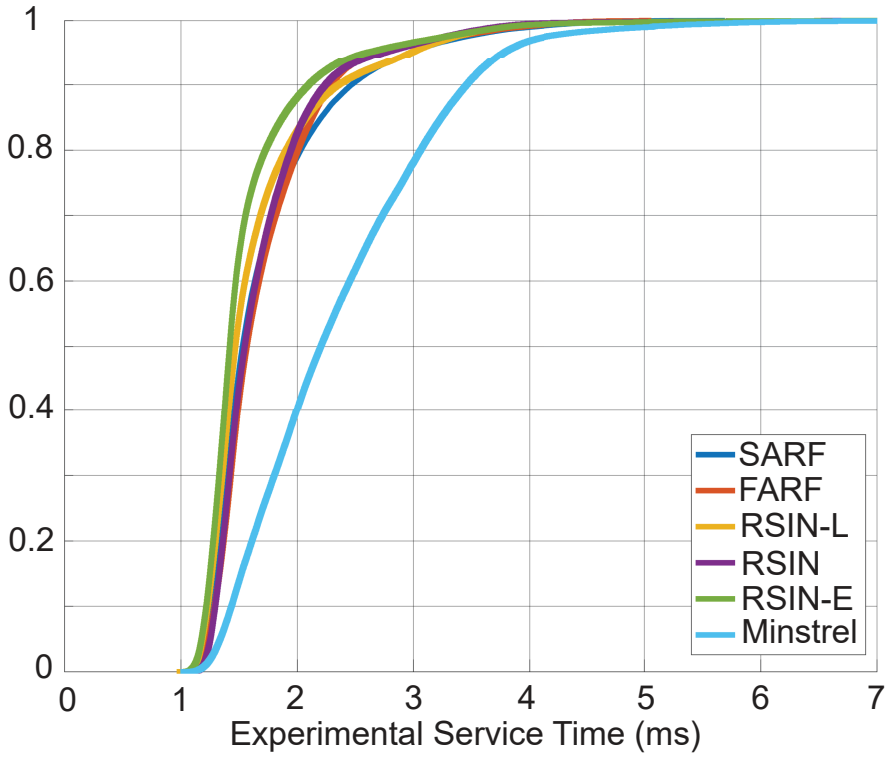

Figure 4.8: ECDFs for the comparison of the RTT when rate adaptation is enabled.

Table 4.5: Statistics of the Round Trip Time and Impact of the RA techniques

	Round Trip Time		
Algorithm	**Mean**	**Std. deviation**	**Impact** [%]
SARF	1.69607 (ms)	0.978333 (ms)	0.1836
FARF	1.6906 (ms)	0.965661 (ms)	0.2134
RSIN-L	1.61245 (ms)	0.927467 (ms)	0.2423
RSIN	1.66003 (ms)	0.891203 (ms)	2.7478
RSIN-E	1.53468 (ms)	0.789313 (ms)	0.7250
Minstrel	2.26982 (ms)	1.47906 (ms)	0.1143

such as RSIN and RSIN-E that showed longer execution times than the other ones. This result on the one hand confirms that the adoption of the rate adaptation algorithms is advantageous whereas, on the other hand, it shows that communication times are predominant over elaboration times. As a last consideration, it is also possible to notice RSIN and RSIN-E present also lower values of standard deviation. This is a good result, pointing to increasingly deterministic communication protocols.

4.1.5 Minimum Cycle Time for a Wi–Fi based Real–time Industrial Network

In a final session of tests, I carried out a set of simulations to investigate the impact of the execution times on the Minimum Cycle Time (MCT), a typical performance index for industrial networks [22], [137]. The MCT has been introduced for networks based on a cycle (that are widespread in the industrial scenario) in which a master device regularly polls a set of slaves. In such a kind of configurations, MCT is defined as the minimum time employed by the master to subsequently poll all the slaves.

I referred to industrial network configurations typically deployed at the lower levels of factory automation systems, such as production islands, that comprise a controller and a limited number of sensors/actuators [138]. Also, the parameters selected for this test are those listed in Table 4.1. For the sake of simplicity, aperiodic traffic has not been considered. In order to execute realistic simulations, I took into consideration the results of the previous measurement sessions. In particular, the measured MAC layer execution times were introduced in the simulation model of all the involved nodes. Moreover, from the communication point of view, I adopted an approach similar to that presented in [139] and [140]. Transmission and Reception correlation matrices are calculated to simulate a multi-path MIMO channel. Specifically, the “F” channel model, proposed by the IEEE 802.11 Task Group n, has been used. Such model is targeted for industrial environments. The simulation setup comprised one master and a variable number of slaves. The polling of a slave, as described in Figure 4.2, starts with a data request packet issued by the master and ends with the reception of the response data from the slave. In the simulations, I considered two rate adaptation algorithms among those addressed so far, namely RSIN and RSIN-L. Furthermore, as

a basis for reference, I plotted the behavior of the MCT for the (purely theoretical) case in which the execution time of the rate adaptation algorithm was not considered. This is referred to as "No–ET". The settings of the network were those presented in Table 4.1.

The behavior of the MCT versus the number of nodes is provided in Figure 4.9 that reports the mean values (continuous lines) and the variation range relevant to the 5th and 99th percentiles, respectively (dotted lines). The statistics are reported in Table 4.6. To ensure good readability the figure does not comprise RSIN-L that, however, has a behavior very close to that of No–ET. As can be seen, although the adoption of RSIN leads to a greater MCT, with respect to No–ET, its impact is rather limited. As an example, for the case of 20 nodes, it results 2.94%. These outcomes are confirmed by the statistics of MCT, which also show that the variability introduced by the rate adaptation algorithm on the MCT is negligible. As a final remark, it may be observed that, practically, the MCT increases linearly with the number of nodes. This is not surprising, because the main cause of non–linearity is represented by packet retransmissions during the polling of slaves, that introduce random backoff times, with the consequent negative impact on the MCT. A packet needs to be retransmitted either when it incurs in a collision or when it is lost for other communication impairments. However, with the adopted Master–Slave protocol, collisions are negligible (slaves can not transmit contemporaneously) and so are the consequent packet retransmissions. Moreover, packet retransmissions due to other causes are limited by RSIN, that cleverly selects transmission rates able to ensure the best packet transmission success probability.

Table 4.6: Statistics of the MCT and Impact of the RA algorithms.

Nr. Nodes	**RSIN**		**RSIN–L**		**No–ET**	
	Mean (ms)	**Dev.** (ms)	**Mean** (ms)	**Dev.** (ms)	**Mean** (ms)	**Dev.** (ms)
5	8.5064	0.4198	8.3022	0.4241	8.2566	0.4274
10	16.9741	0.5772	16.5751	0.5879	16.4748	0.5883
20	33.9502	0.8226	33.1381	0.8314	32.9521	0.8386

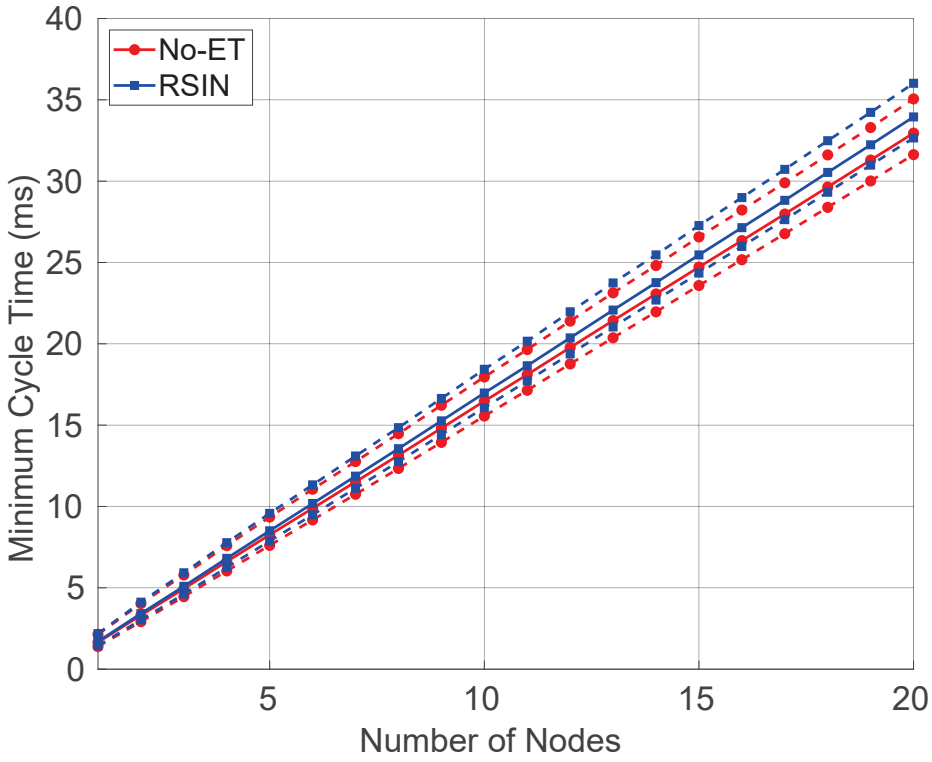

Figure 4.9: Minimum Cycle Time for a Prototype Industrial Network.

Chapter 5

Optimizing LoRaWAN for IoM

Wi-Fi networks are surely attractive within the IoM concept, as they are high-performing and speed networks. As a matter of facts, there are several industrial applications where high data rates are not required, but long communication range and low energy consumption are key requirements. For example, a useful test-case is presented in the following. The ADMIN-4D project [141] foresees the development of 3D printers of huge dimensions, devoted to the printing of dumb-based artifacts for various applications. In this context, humidity and temperature must be carefully tracked during both the production phase of the artifact, and in the final deployment site. In particular, during production data coming from the artifact are used to control the process (e.g. sent to a PLC) as depicted in Figure 5.1. Moreover, data from both the production and in the final deployment phases are collected in the cloud to be used for offline analysis.

In such application, low data rates are acceptable, but long range (especially when the artifact is in the final deployment site) and battery consumption are surely key issues. The latter is an important parameter, as once the sensors are embedded in the artifacts are not accessible anymore. For this reason, a LoRa network has been chosen for the project. Indeed, by comparing Figures 5.1 and 3.4 it is possible to see that the considered network architecture is exactly the same. An experimental campaign to test different kind of batteries and their lifetime has also been conducted. With a sampling

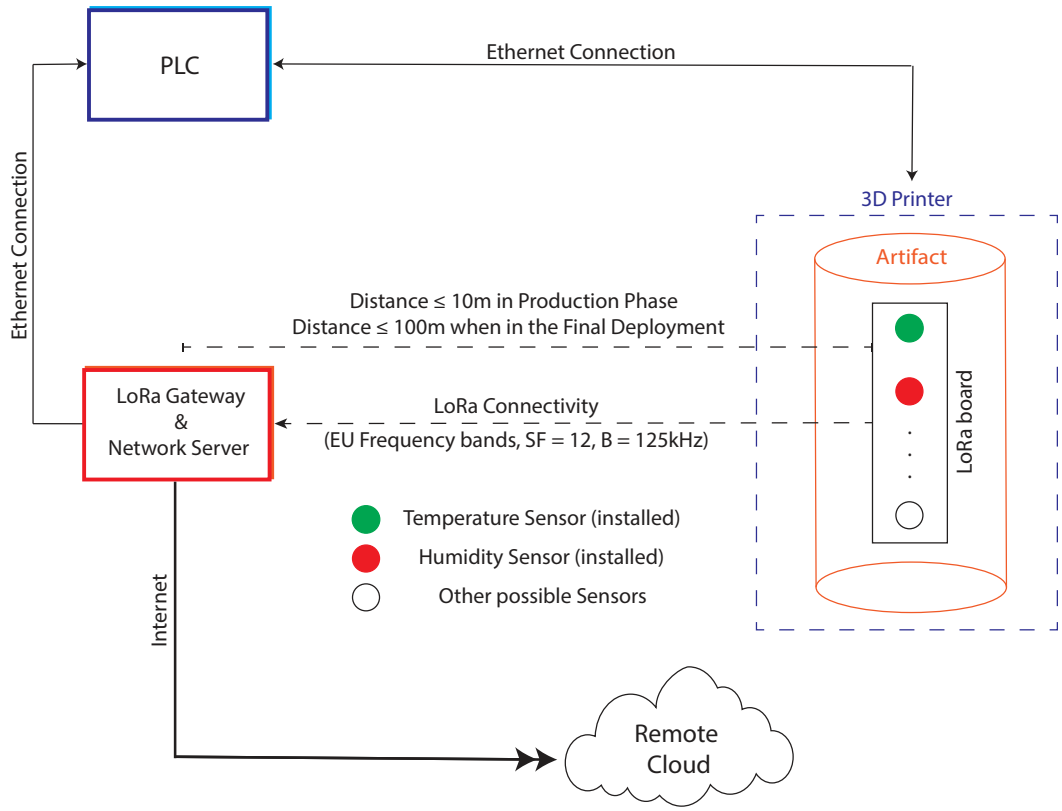

Figure 5.1: Architecture of the ADMIN-4D test case.

period of 5 minutes, it was possible to obtain a battery lifetime of approximately 18 days and 60 days by using respectively widespread Lithium-ion (Li Ion) and Lithium Thionyl Chloride (LTC) batteries.

By carefully analyzing this meaningful test case, it is possible to draw some conclusions, and challenges in the application of LoRa network for industrial systems and, generally speaking, in the I&M field.

1. First of all, common requirements coming from the IoM need to be investigated and optimized, as already discussed in Section 1.2. The number of correctly sent packets and communication delays must be investigated.

2. Battery consumption must be decreased. Indeed, for the considered application, the battery lifetime is enough but generally speaking, and increasing the sending frequency, an optimization of the energy consumption is needed.

3. To generalize, and apply LoRa networks in a wider area of applications, the already discussed (in Section 3.4.2) DC limitation must be addressed, to increase the data rate.

5.1 Parameter Adaptation Using RL

In order to suitably address the aforementioned issues, it is possible to adjust the transmission parameters introduced in Section 3.4.1 can be adjusted. In practice, this parameter adaptation is something similar to the already discussed RA for Wi-Fi networks. In the following, a novel parameter adaptation strategy based on RL is presented and compared to the default parameter strategy of LoRa, namely ADR (Section 3.4.3). Results obtained in this section demonstrate that the proposed RL strategy achieves a better trade-off between the number of correctly sent packets and the energy consumption.

5.1.1 Formalization

As already stated, the development of performing and accurate industrial IoT measurement systems, poses several challenges in terms of correctly delivered packets, under strict timing requirements. In this context, the DER that has been already introduced in Eq. (1.2), becomes a key parameter for this study. It is worth observing that all the following RL strategies follow the approach introduced in Section 2.9, with particular reference to Eq. (2.9) and Eq. (2.10).

Moving from the aforementioned observations, a RL–based parameter adaptation policy has been proposed, namely RL-based Rate Adaptation Algorithm (RRAA). The latter, bases its behaviour on a suitable MDP, where states, actions and rewards are defined as follows. First of all, states are defined as couples of DER and SNR values. In particular, both DER and SNR variation intervals of the UpLink messages are discretized to suitably describe all the possible communication conditions. Ten different equally spaced DER levels have been identified, ranging from 0% to 100%. Moreover, after a careful analysis of the behaviour of the simulator, 12 SNR levels ranging between 0 and $60dB$, and one level to represent $SNR > 60dB$, have been recognized. The proposed discretization allows to describe the system environment by means of 130 different states. The SNR becomes, in this context, a key parameter, able to suitably represent the channel's condition, depending on both the distance between node and GW and the interference with other devices. For this reason, the Q table resulting from the training activity can be shared between all nodes, thus developing a smart, mobile and plug and play measurement system. Secondly, Actions are defined as couples of TP and SF, whose possible values have been already presented, respectively, in Eq. (3.4) and Eq. (3.5). By combining them a total number of 30 actions can be identified.

After a trial and error phase, rewards for RRAA have been defined as per Eq. (5.1), to suitably optimize the total DER.

$$r = \frac{(DER_t - DER_{t-1}) * SNR_t}{SF_t} \tag{5.1}$$

Specifically, higher rewards are given to actions resulting in higher DERs, SNRs

and lower SFs, i.e. higher data rates. The latter, in particular, aims at accomplish a trade–off between communication robustness and low transmission delay. Indeed, the DER, the SNR and the SF are monitored from the environment after an action is taken. It is worth observing that, if the algorithm is run in a static network configuration and the same interference model, the considered variables varies mostly due to the chosen action. By combining them as per Eq. (5.1), higher rewards are given when both the DER and SNR increase. By only taking into account the numerator in Eq. (5.1), the algorithm is always choosing the most robust communication possible, that will result in higher communication delays. For this reason, I decided to divide by the SF, to give higher rewards to lower SFs, that in turn increase the data rate. This adaptive strategy does not take into account the energy consumption, being suitable for applications where the battery lifetime of the sensors is not critical. On the contrary, there are several industrial applications where measurement systems must also accomplish to battery lifetime requirements, as the aforementioned one. In this context, sensors become unreachable, thus underlining the importance of the battery lifetime. For these reasons, the battery consumption must be taken into account. The novel RL algorithm, namely Green RRAA (GRRAA), foresees to modify the rewards, to take into account the TP. The new reward function is presented in Eq. 5.2.

$$r = \frac{(DER_t - DER_{t-1}) * SNR_t}{\beta * SF_t * TP_t} \tag{5.2}$$

It is worth observing that, the β value can be used to suitably tune the RL adaptation policy. In particular, the higher β, the higher the importance of SF and TP in the rewards. In this situation, much attention is given to the energy consumption, as the algorithm tries to use lower TPs and SFs. Actually, the lower the SF, the higher the data rate, the lower the transmission time, thus reflecting in a lower energy consumption and transmission delay. On the contrary, the lower β, the higher the importance of the DER. In conclusion, α and γ values of Eq. (2.9) are set, respectively, to 0.1 and 0.7, while β is set to 10.

The training activity starts initializing the Q_{table} with null values. When a downlink message is received from a specific node, the DER and the SNR values are used to

determine the new state while the rewards are calculated using Eq. (5.1). Then, the new Q-value is calculated with Eq. (2.9). Once the Q-table has been updated, the new action is determined exploiting the ϵ-greedy algorithm Eq. (2.10), and the SF and TP values are updated.

5.1.2 Simulation Set-Up

LoRaEnergySim (LES) [142], a Python-based simulation framework, has been used to properly evaluate the behaviour of the three algorithms (RRAA, GRRAA and ADR). In particular, LES allows to suitably simulate collided and weak packets. Moreover, as the simulator already implements the ADR adaptive technique, it is particularly suitable to compare ADR with the proposed adaptive strategy in terms of DER. In addition, an accurate energy consumption model is used by LES, thus allowing to achieve meaningful results in terms of used power.

The LES provides objects for modeling one or more "Nodes", a single "GW" and a suitable "Air Interface". The latter one suitably models the propagation bahavior, the SNR, and the collisions. In this way, the communication between several Class-A LoRa nodes has been suitably simulated. As a first step, two simulation scenarios have been considered. Figure 5.2 represent the simulated network architecture.

As can be seen in Figure 5.2, nodes are placed in circumferences centered on the GW, respectively with a radius of $100m$ and $3950m$. In both scenarios, a total of 20 nodes has been used. This particular choice has been made to obtain simple but meaningful environments allowing a clear analysis and comparison of the transmission parameters selected by the three algorithms, depending only by the distance between the node and GW. In the depicted scenarios, the network backend, after the reception of N_{step} consequent uplink messages, sends a downlink message communicating the RP_t value of the specific node. The latter information allows the DER calculation as per Eq. (1.2).

Table 5.1 resumes the simulated environments setup.

It is worth noting that, in Table 5.1, the duty cycle is set to 1%. Then, the inter-message delay has been set to the highest value, to suitably respect the imposed duty cycle also when the data rate is the lowest (i.e. SF12). In the next section, simulation

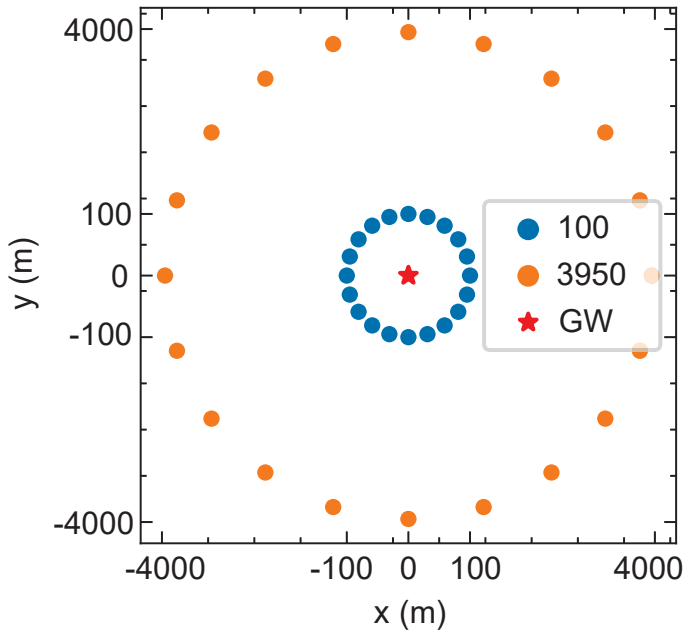

Figure 5.2: Locations of the nodes

Table 5.1: Simulation setup

Parameter	**Value**
Number of nodes	20
Area of the simulated environment	$64Km^2$
Total UpLink packets per node (N)	1000
Period of confirmation downlink message (N_{step})	5
Payload length ($Byte$)	50
Duty Cycle (%)	1

results are presented and discussed.

5.1.3 Results and discussion

After an extensive training phase, it was possible to test the adaptation algorithms, in all the aforementioned scenarios. In particular, the chosen SF and TP values are presented respectively in Figure 5.3 and in Figure 5.4.

The most important observations can be derived from Figure 5.4. In particular, the RRAA algorithm does not optimize the choice of TP, that has been mostly set at the higher values. Clearly, this involves in higher DER compared to ADR, that optimizes also the energy consumption. On the contrary, the GRRAA better optimizes the usage of TP. By comparing the behavior of GRRAA and ADR, it is possible to underline that the network presents different behaviors in the two considered scenarios. Indeed, ADR chooses mainly the lower TP when the distance from the GW is very low, while setting high TPs when dealing with more challenging scenarios. Actually, GRRAA balances better the TPs in both scenarios. By analysing the SF, it is also possible to underline that, by highering the β value in Eq. (5.2) it is still possible to lower the energy consumption, and also the transmission time. Indeed, GRRAA presents higher DERs because it makes use of higher SFs (especially when the distance from the GW is high), compared to ADR. This leds to higher transmission times and energy consumption. Moreover, comparisons between the three algorithms in terms of DER and Power spent per Packet (PP) are depicted respectively in Fig. 5.5 and 5.6. Furthermore, the mean DER and PP values are listed in Table 5.2.

Results in terms of both DER and PP, are totally consistent with the analysis made above. When the distance from the GW is too low, all the algorithms are capable to correctly deliver packets, and ADR performs better in terms of energy consumption. On the other side, GRRAA performs definitely better in terms of PP, while maintaining high DERs, when the distance from the GW becomes higher. From Table 5.2, it is possible to draw some last observations. Firstly, all the algorithms perform better in terms of DER when the distance from the GW is lower, thus underlining the correctness of the proposed algorithms. Moreover, the power spent to send a single packet grows with the distance from the GW. Actually, the GRRAA provides quite similar energy

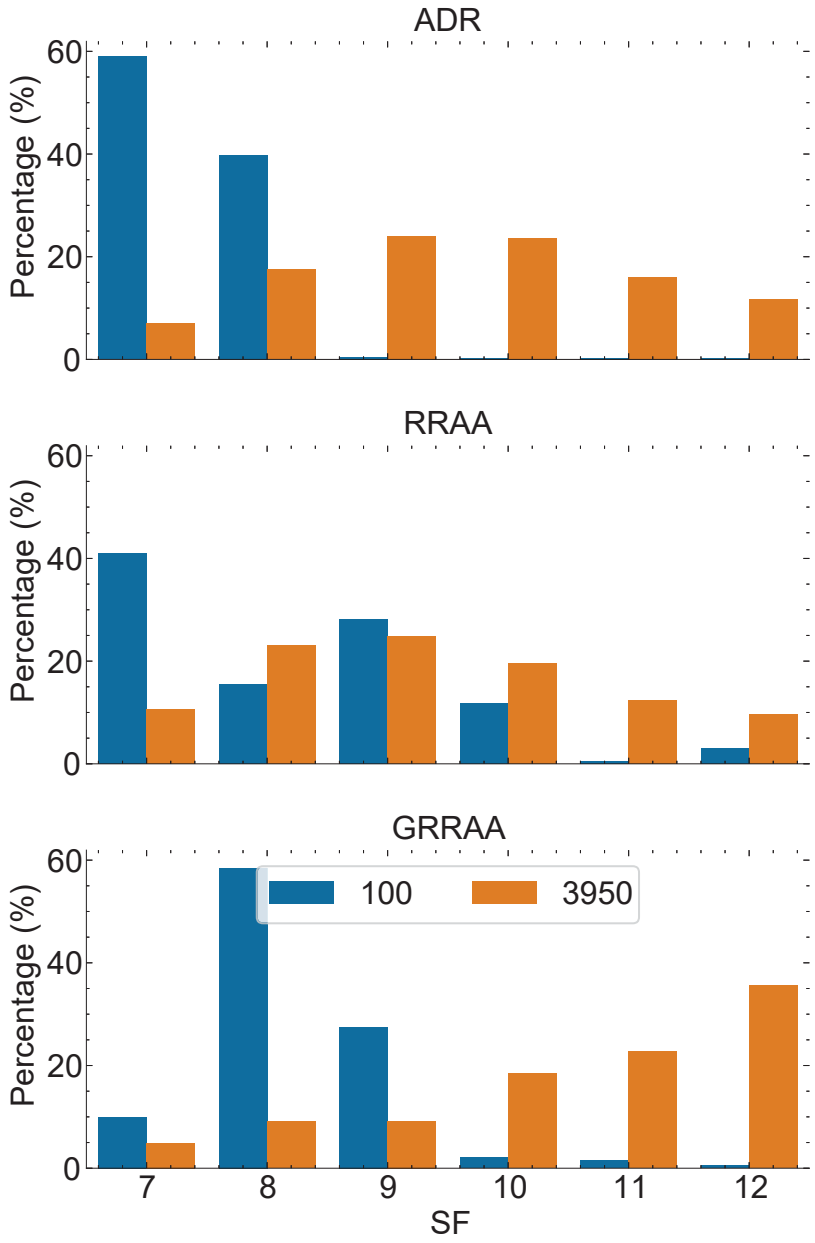

Figure 5.3: Comparison of the SF in two different scenarios with ADR, RRAA, and GRRAA

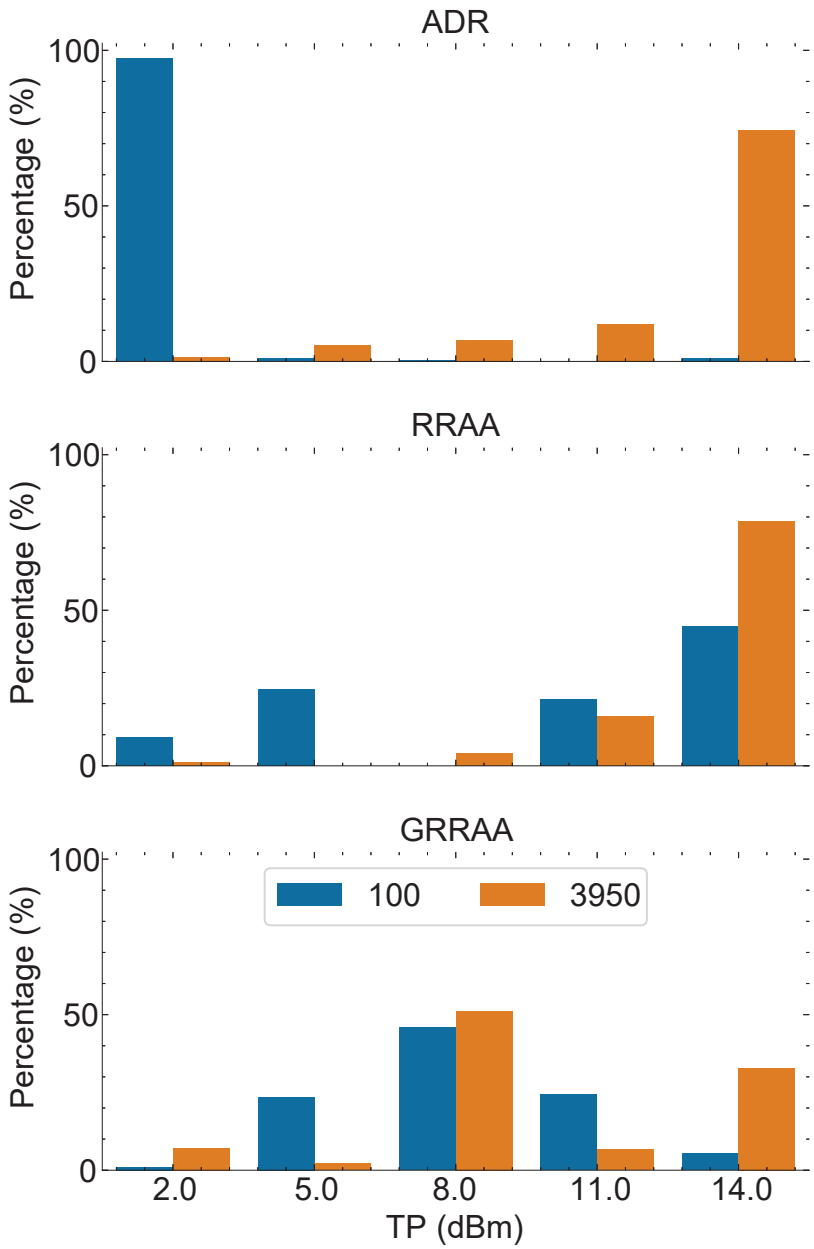

Figure 5.4: Comparison of the TP in two different scenarios with ADR, RRAA, and GRRAA

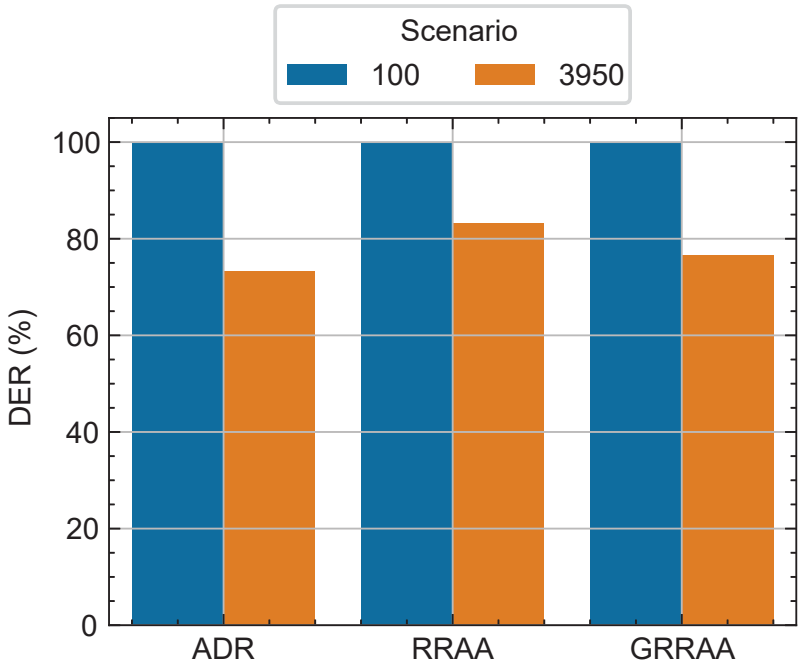

Figure 5.5: Comparison of the DER in two different scenarios with ADR, RRAA, and GRRAA

Table 5.2: Comparison of the Power per Packet in two different scenarios with ADR, RRAA, and GRRAA

Name	SimNo	DER	PWR per PKT	PWR over DER
ADR	100	99.88	182.08	1.82
ADR	3950	73.25	233.33	3.19
RRAA	100	100.00	216.65	2.17
RRAA	3950	83.12	236.69	2.85
GRRAA	100	99.77	199.37	2.00
GRRAA	3950	76.60	215.12	2.81

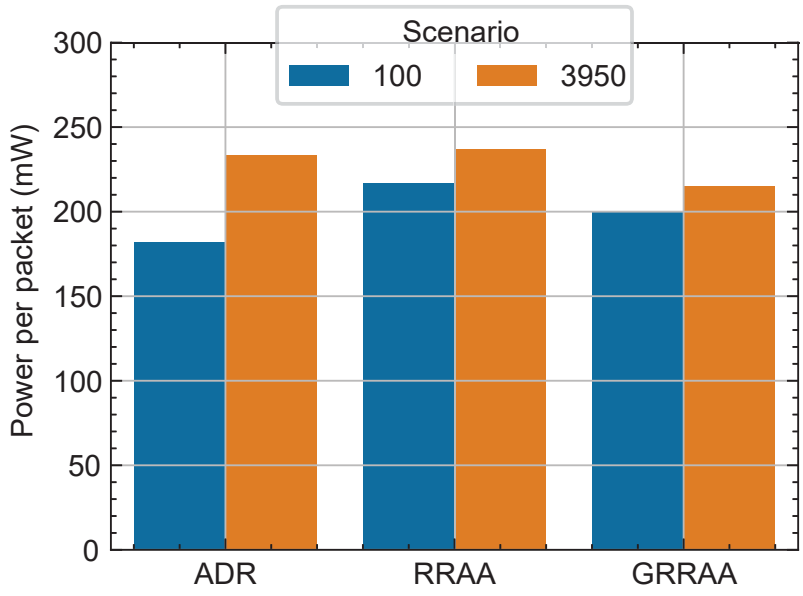

Figure 5.6: Comparison of the Power per Packet in two different scenarios with ADR, RRAA, and GRRAA

Table 5.3: Comparison of the Power per Packet on the random locations scenario with ADR, RRAA, and GRRAA

Name	DER	PWR per PKT	PWR over DER
ADR	94.04	192.13	2.04
RRAA	98.94	210.75	2.13
GRRAA	98.97	198.19	2.00

consumption values compared to ADR, while performing definitely better in terms of DER. From a comparison between RRAA and GRRAA, it is possible to underline that the GRRAA performs better in terms of energy consumption, while maintaining similar levels of DER. Summarizing, both the simulation setup and the algorithms implementation are coherent with the theoretical behavior of the network, and the proposed approach is solid and performs better than ADR. Finally, I also considered a scenario where 60 nodes are randomly placed in a $1km^2$ area, with a central GW. The results obtained in such scenario, are presented in Table 5.3.

Actually, the results are encouraging as they confirm the previous ones, allowing to properly generalise the discussion made above.

5.2 Overcoming DC limitation in LoRa networks

In Section 3.4.2 I already underlined that one of the big limitations on the application of LoRa networks for IoM is the DC limitation. In Section, hybrid channel access mechanisms are proposed, aiming to increase the data rate, thus decreasing the delay between the packet generation and the reception. The latter parameter, will be particularly taken into account, and the results will confirm that the proposed approaches decrease the delay and increase the packet delivery rate.

5.2.1 MAC strategies

Several works in the literature have considered the design of different MAC strategies, instead of pure-ALOHA. In fact, a listen-before-talk solution can be effectively

implemented by exploiting the Channel Activity Detection (CAD), already present in all the LoRa radios to detect the presence of incoming preamble chirps [143]. In [144], the authors analyzed several medium access strategies from energy and reliability perspectives, wherein the widespread Carrier Sense Multiple Access (CSMA) listen-before-talk channel access methodology has also been taken into account. In CSMA, a device performs the channel sensing before a packet transmission and proceeds with the data communication only if the channel is sensed idle, thus avoiding collisions among devices. In [143], the authors pointed out that CAD procedure, provided by LoRa devices, can effectively be used for channel sensing. Although this feature is originally designed for energy-efficient preamble detection, authors demonstrated a 95% accuracy in detecting the channel occupancy. Moreover, the authors of [145] estimated the CAD procedure duration, depending on the specific SF. Despite these encouraging results, note that the relationship between the CAD procedure accuracy and the specific surrounding environment (e.g disturbances, obstacles, etc) is still unclear.

5.2.2 Channel Access Designs

In this section, the design of hybrid ALOHA-CSMA medium access strategies to overcome the DC limitation of the LoRaWAN specification. The first solution considers using CSMA in one of the three mandatory channels, with ALOHA in the other two. I refer to this technique as a hybrid ALOHA-CSMA (hybAC) strategy. All the scenarios considered in this Section are based on a basic implementation of CSMA. Between the transmission of a packet and the subsequent one, the device waits for a Distributed Inter-Frame Space (DIFS). Then, the channel sensing is performed, and the CAD timings are added. If the channel is free, the device can start the transmission. If the channel is occupied, the device waits a random time between 0 and a predefined Contention Window (CW), before retrying the transmission. A maximum number of transmission attempts can be done, before considering the packet as lost. The hybAC strategy considers sending the measurements directly to the GW using two available ALOHA channels when possible. The third channel, where measurements are sent through the CSMA channel access mechanism, gives the possibility to send packets during the ALOHA channels' unavailability due to the aforementioned DC limitations.

The second solution investigates a hybrid ALOHA-CSMA strategy with Relaying capabilities (hybACR). The idea is to enable the devices with high inter-message delay to transmit packets by using CSMA to the neighboring devices, which may still have DC left for transmission. These “relays" in turn, transmit the message to the GW using ALOHA. In this way, the nodes in critical situations (i.e., having no DC left to transmit) can ask for help from neighbors, without the need to sense the whole area, but only a limited geographical space around them. An idea could be to optimize the sensitivity threshold to reduce the sensing area. I will refer to this connectivity mode as the Device-to-Device (D2D) communication or relaying strategy. There are several contributions in the literature addressing the LoRa D2D capability. Actually, a comprehensive review can be found in [146], and it is mainly used to provide connectivity to the End Devices out of the reception range of the GW. The authors in [146] also highlighted that all the relaying End Devices must have the capability to simultaneously receive packets at different SFs. Therefore, for the considered application, it is important to consider multi-channel devices, thus allowing the reception of packets from stations with different SFs. A conceptualization of the proposed approach is depicted in Figure 5.7.

Figure 5.7 represents an area served by a central GW and several associated End Devices. The red circles represent End Devices with no more DC left, i.e., they have consumed the allowed DC quota, as depicted in the figure with zero slots in the DC queue. In this case, the red devices can use the available DC of the black devices, by transmitting their frame in D2D mode using CSMA mechanism. Red nodes, when needed, can send a broadcast *help_request*, asking to use other nearby stations as relays. It is worth noting that the area in which one station can sense the channel and send *help_requests* through CSMA, strictly depends on the chosen energy detection threshold; thus, it becomes a tunable parameter of the whole approach. The problem is, in fact, twofold. Stations with DC left must choose one (or a set) of other stations to serve as a relay, and send to them an *help_ack* comprising information about the DC and the SNR. This information will be used then by the red devices to choose the best station as a relay, among the ones that acknowledged *help_requests*. In substance, for relay selection, an Request To Send (RTS)/Clear To Send (CTS) mechanism is

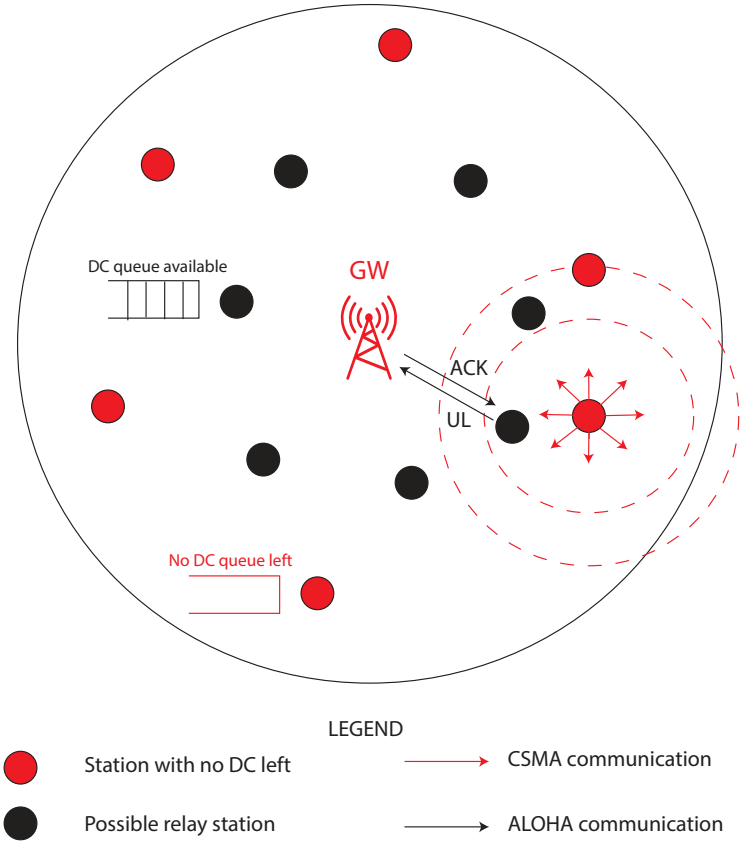

Figure 5.7: Conceptualization of the proposed approach.

enabled to pair stations with no DC queue left with relays. By analyzing Figure 5.8, it is possible to understand how the RTS/CTS works.

It is also worth observing that the RTS/CTS mechanism is able to handle the so-called *hidden terminal* issue. In particular, each station at each time checks if, following the duty cycle limitations, it has the possibility to transmit a packet. If so, and no packet is ready for transmission, the radio switches to receiving mode, waiting for the *help_request* of another node. If more than one RTS is received, the station chooses to relay information coming from the first node that asked for help. Similarly, if more CTSs are received from a node requesting help, the first station sending CTS is used

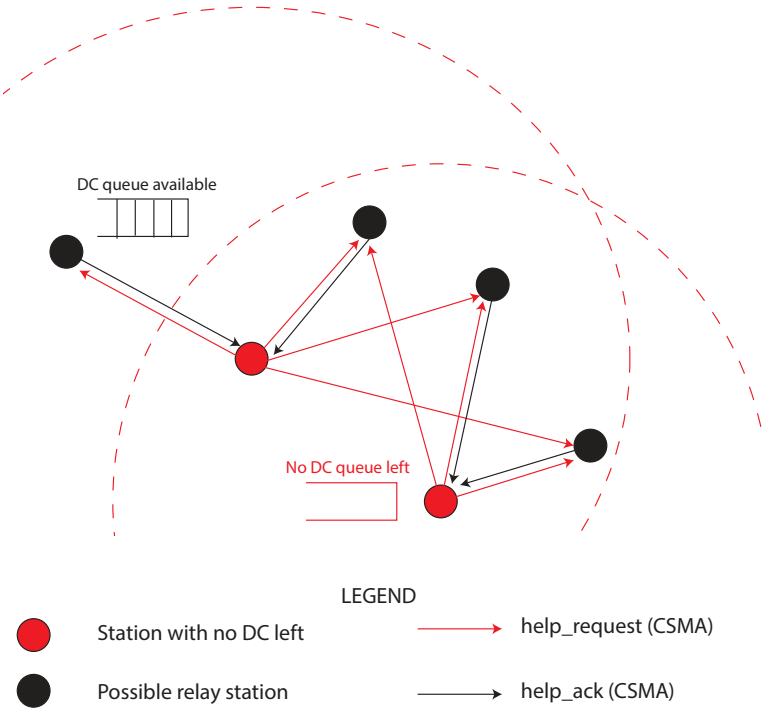

Figure 5.8: RTS/CTS mechanism.

as a relay.

It is worth observing that the problem I want to solve takes place on two different levels. The network behavior is strictly related to the specific set of selected communication parameters. In particular, given Eq. (3.6), Eq. (3.7), Eq. (3.3) and Eq. (3.9), it is possible to notice that BW, CR, and SF strictly affect the inter-message delay. For this reason, the parameter adaptation in this context is strictly related not only to the desired communication performances but also to the possibility to relay and schedule all the needed data traffic. Despite this, it is reasonable to first consider separately the adaptation of the communication parameters and the traffic schedule. Communication parameters can be at a first glance chosen, aiming at a high-performing communication, based on the distance of the specific node to the GW. In particular, it is possible to assume that the stations more distant from the GW have a higher SF, also if this

behavior depends on the specific adaptation strategy employed, as outlined previously. In this way, moving from the center of the network to the border, the SF increases, and the inter-message delay increases.

5.2.3 Simulation Assessment

A suitable simulation environment has been set up, to evaluate the strategies proposed in the previous section. As in the previous Section, the LES simulator [142] has been exploited, by adding a basic CSMA implementation and enabling D2D communication. In this study, the CAD procedure has been considered enough effective to derive the channel occupancy, so that the information about the channel status has been simply shared between nodes in the simulator. Despite this, the timings related to the CAD procedure have been properly taken into account. In this regard, the CAD procedure timings can be derived from literary works, following an approach similar to [147].

By exploiting the aforementioned simulator, a network comprising a central GW and End Devices at different distances are tested. The network topology, together with the communication setup, is depicted in Figure 5.9.

I evaluated both the hybAC strategy and the hybACR strategy, comparing them with the pure-ALOHA strategy in terms of correctly delivered packets and End-to-End (E2E) delay. In this context, E2E comprises all the communication delays and also the time packets wait in the queue for being transmitted. This definition of E2E is particularly meaningful in this context, to evaluate the aging of the measurements when they are received by the GateWay.

5.2.4 Results

The network already depicted in Fig. 5.9 is simulated by using the communication parameters as given in Table 5.4.

The conducted simulations lasted 2 hours, where Poisson-distributed traffic is generated, with a total number of 1080 packets per node. In these 2 hours, the E2E has been monitored only for half an hour, as its value (for ALOHA algorithms) quickly

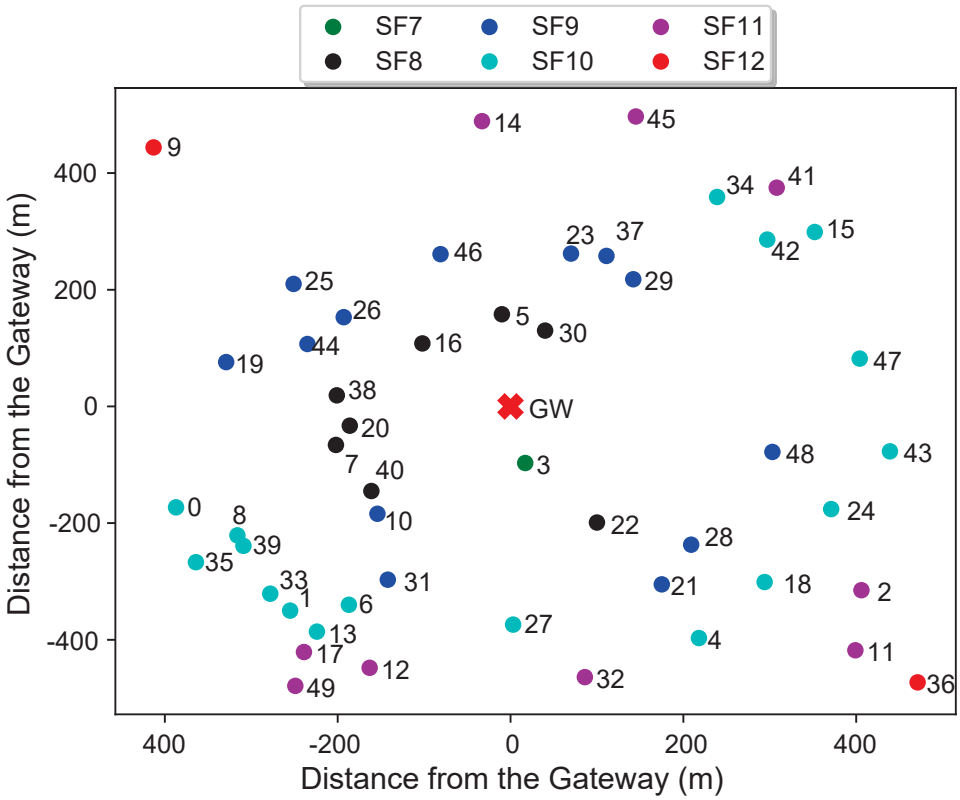

Figure 5.9: Network Setup.

diverge. This behavior is given by the fact that at a certain point ALOHA can not serve any more packets in the queue of nodes with high SF. For this reason, monitoring the delay for more than half an hour is not meaningful.

The first simulation objective was to test the pure-ALOHA channel access mechanism in LoRa, using all three mandatory channels. The average number of sent packets, for the different SFs, is summarized in Table 5.5.

Notice that the results obtained are totally comparable with the theoretical ones (Table 3.3). Indeed, as expected, for SF 7, 8 and 9 the DC limitation has no effect as the number of generated packets are much less then the number of packets that can be served. For the other SFs, if we consider three channels, the number of sent packets

Table 5.4: Simulation parameters.

Parameters	**Symbol**	**Value**
Bandwidth	BW	125 kHz
Payload	PL	5 bytes
Transmit power	TP	14 dBm
Duty cycle	DC	1 %
Simulation time	T	2 h
Packet generation rate	λ	0.15 pkts/s
Total measurements per node	pkt_n	1080

Table 5.5: Average number of sent packets with a pure-ALOHA channel access mechanism.

SF	**Average sent packets**
7	1075
8	1078
9	1066
10	695
11	348
12	228

are below the theoretical ones. As a matter of fact, in the simulations there are more delays taken into account, trying to mimic the behavior of a real hardware platform. Moreover, this setup is particularly interesting as nodes with SF 7 and 9 can send more messages than the planned ones, while stations with SF 12 can not send all the packets.

Afterwards, the hybAC strategy has been evaluated. The statistics of the sent and correctly delivered packets are reported in Table 5.6.

Obviously, in a low-traffic scenario, the hybAC scenario and the pure ALOHA one perform similarly, according to what is already outlined by authors of [148]. On the contrary, the simulated scenario has been suitably overloaded to demonstrate the

Table 5.6: Percentage of sent and received packets with respect to the generated ones, pure ALOHA and hybAC.

SF	**Aloha SP**(%)	**Aloha RP**(%)	**hybAC SP**(%)	**hybAC RP**(%)
7	97.04	97.04	97.04	97.04
8	95.65	92.60	95.74	89.44
9	98.55	95.38	98.92	92.22
10	64.38	57.15	97.93	92.27
11	32.22	30.21	97.53	93.95
12	21.11	21.11	99.44	98.70

effectiveness of the usage of a Listen Before Talk strategy. The statistics reported in Table 5.6 have a big impact also in the time between the measurements generation and the reception of the packet from the gateway. The cumulative distribution function of the aforementioned delay, both for the pure-ALOHA and hybAC methods are reported in Figure 5.10. As a matter of fact, the CSMA methodology drastically decreases the E2E delay, when in presence of high-traffic networks. This behavior is confirmed also by Table 5.7, where both mean and standard deviation values of the delay are reported.

Table 5.7: Delay between measurements generation and reception: statistics.

Scenario	**Mean delay**(s)	**Standard dev.**(s)
Pure ALOHA	154.24	255.73
hybAC	1.59	2.38

As can be seen, when using a pure ALOHA mechanism, not only the mean value but also the standard deviation are higher than the hybAC methodology. As said before, this strongly impacts on the measurement uncertainty.

Finally, the hybACR methodology has been evaluated. To do so, I used the network deployment of Figure 5.11.

Indeed, a random network will always make the D2D strategy perform poorly.

Indeed, as SFs are tuned based on distance, each node will find as neighbors devices with the same (or similar) SF, thus leading to a non-availability of DC left also on nearby stations. For this reason, I simulated a situation where nodes with lower SFs could help devices with no DC left. A comparison in terms of correctly received packets and E2E are respectively presented in Table 5.8 and 5.9.

Table 5.8: Percentage of sent and received packets with the respect to the generated ones, both using pure ALOHA and hybACR.

SF	**Aloha SP**(%)	**Aloha RP**(%)	**hybACR SP**(%)	**hybACR RP**(%)
7	97.78	94.38	97.72	92.78
12	22.22	22.22	96.67	80.37

Table 5.9: Delay between measurements generation and reception: ALOHA vs hybACR statistics.

Scenario	**Mean delay**(s)	**Standard dev.**(s)
Aloha	25	15
hybACR	1.26	2.4

As can be seen, in such a situation hybACR increased the communication capabilities of the sensor network. It is worth observing that, as before, the E2E has been monitored for half an hour. Here, also in half an hour, the delay is very big and this effect is only due to the two SF12 stations. This strategy is particularly useful when a low number of nodes are in a critical situation (low SF, presence of obstacles that lower the transmission range, etc) and suitable devices with lower SF can be used as to relay the information to the GW. This will decrease the occupancy of the channel, compared to enabling direct CSMA communication to the GW. Moreover, this strategy will be particularly appealing if we take into consideration the D2D capability when dynamically adapting the SF.

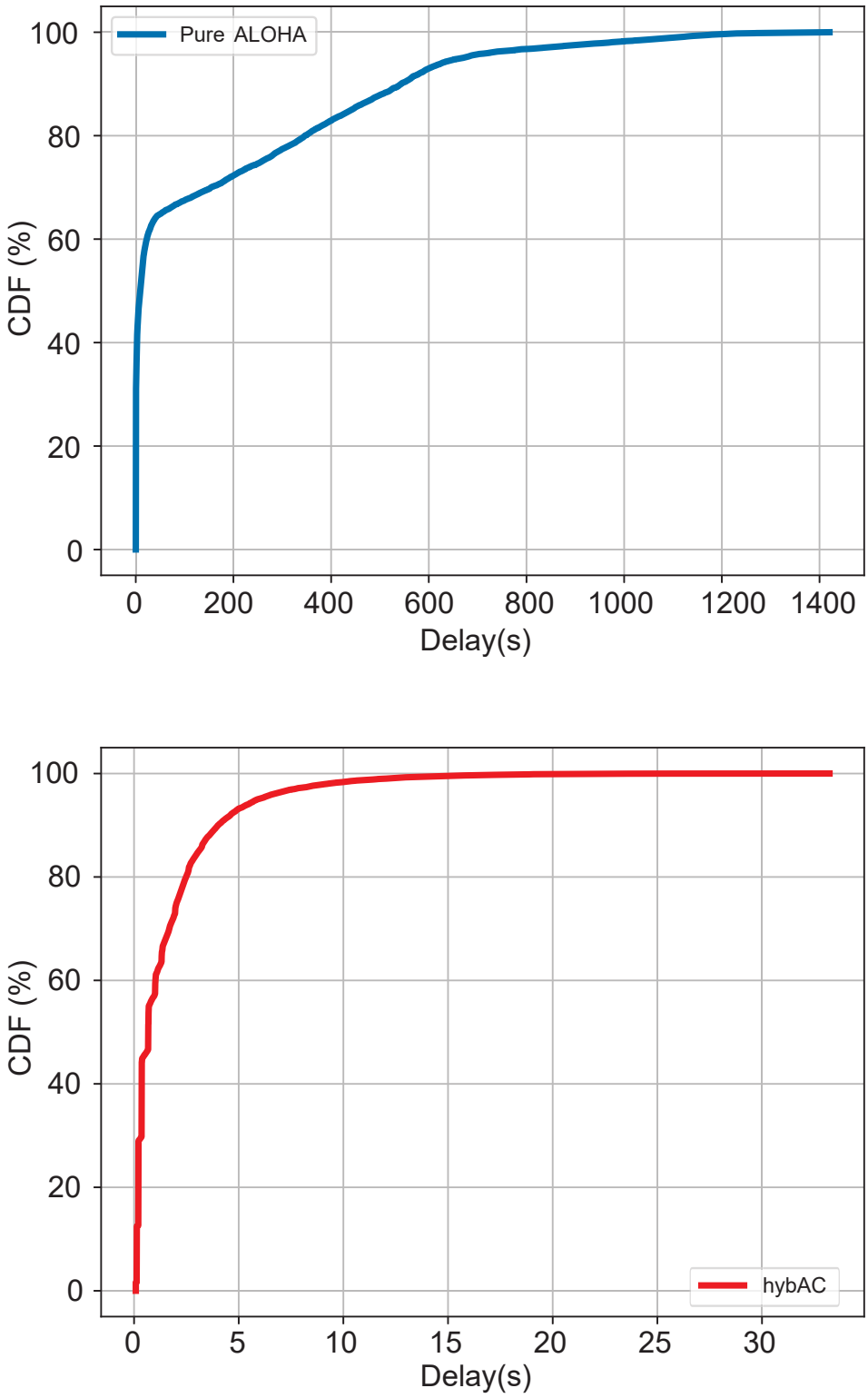

Figure 5.10: Pure ALOHA and hybAC strategies delays CDF.

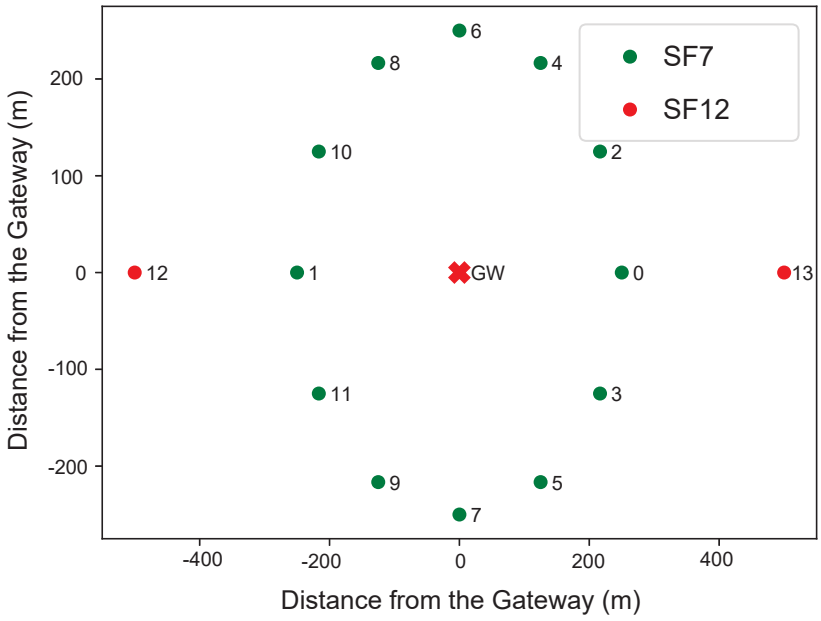

Figure 5.11: Network used to simulate the hybACR.

Chapter 6

ML - Based Post Processing Techniques

In the previous chapters, the work focused on the optimization of wireless connectivity devoted to measurement transmission within the IoM concept. In this chapter, I analyze the possibility to exploit ML techniques to derive measurement from images. As already stated, AI techniques are a key enabler of the IoT, although if some research challenges must still be investigated. ML - based post processing techniques have been evaluated by using as a test-case a meaningful example of Vision - Based IoM, that is presented in Section 6.1.

6.1 A Van Herick Vision Based Measurement System

Glaucoma and other eye diseases are affecting more and more people in the last few years. Prevention is essential to avoid the progression of that disease, but in some cases, the screening exams are invasive or quite expensive, and it is not possible to periodically monitor the eye condition [149]. It has been demonstrated that people with a narrower Anterior Chamber Angle (ACA), i.e. the iridocorneal angle, are more vulnerable to the most aggressive form of glaucoma, the Primary Angle Closure Glaucoma (PACG) [150]. Nowadays, the gold technique used for ACA measurement is the gonioscopy,

but it is invasive and requires high medical skills [151]. Among the various assessed techniques, one of the most interesting is theVan Herick (VH) maneuver, which exploits the correlation between the thickness of the cornea and the ACA [152]. The ratio between these two thicknesses represents the width of the ACA, hence allowing to detect the PACG. VH approach requires a slit lamp, illuminating the limbus with a 60 degrees angle between the light source and the eye optical axis [153]. Despite the simplicity of the approach, any PACG diagnosis derived from the estimation of the ACA with the VH technique must be performed by direct observation of an expert ophthalmologist with the help of a traditional slit lamp and a microscope. It appears then clear that, necessarily, this type of diagnosis is intrinsically subjective and results are strictly related to the ability and experience of the observing ophthalmologist [151].

In the following, the developed VH instrument is presented, and some ML based post processing techniques are adopted.

6.2 Experimental Setup and Research Goals

The Van Herick procedure used to measure the ACA must be accomplished with specific alignment constraints [151]. It has been shown by authors of [154] that both illumination and observation angles affect the ACA openness assessment. As a consequence, particular attention was paid during the realization of the optical setup for such an experimental evaluation. A schematic diagram of the optical setup used to perform the Van Herick measurement is shown in Figure 6.1.

The optical setup consists of two main devices. Firstly, a digital CMOS camera Basler Dart (daA1600-60uc S-Mount, Basler© AG, Ahrensburg, Germany) is positioned in front of the analyzed eye, aligned with its optical axis. A 16 mm focal length lens (Evetar Lens M12B1618IRM12 F1.8) is also used in the optical setup. Secondly, a LED Digital Light Projector (DLP) (DLP2010EVM-LC, Texas Instruments©, Dallas, Texas, U.S.) was used as the illumination unit, instead of the traditional slit light. The DLP relies on modern micro-mirror technology to project structured light onto a specific target. The illumination unit was placed at a 60-degree rotation angle with respect to the eye and camera optical axis as shown in Figure 6.1. Moreover, the DLP

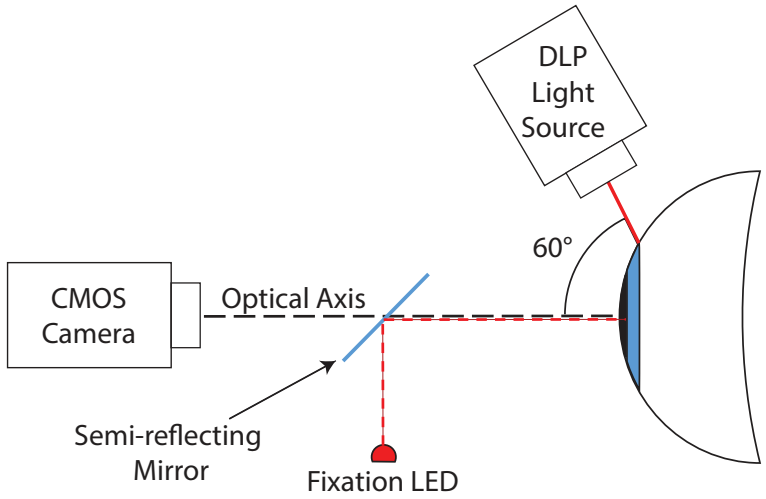

Figure 6.1: Schematic diagram of the optical setup.

has been configured to emit a uniform red slit light that scans the whole surface of the eye under examination. Thus, during the measurement procedure, the emitted slit light scans the eye from the external corner of the sclera towards the nose. This scan takes place during the frames acquisition with the digital CMOS camera. A fixation target, i.e. a small light pointer, was placed on the eye-camera optical axis, through a 45 degrees semi-reflecting mirror, to help the patient to look straight ahead. The patient's head is then placed on a chin rest to guarantee the steadiness and alignment to the optical setup. A single measurement procedure has been designed to perform two entire scans of the eye in 4 seconds, allowing for a total of 120 raw pictures to be collected at 30 fps. The entire system is connected to a Windows® based embedded computer (Lattepanda Alpha 864s) with an Intel® Core™ m3-8100Y processor unit. The computer runs a Python-based Graphical User Interface (GUI) that is in charge of peripheral control (Camera and DLP) as well as real-time image processing. Actually, the set of images previously acquired by the camera are then processed by the ML algorithm, aiming at the identification of the central images.

As previously mentioned, the VH technique foresees the comparison of the depth of the peripheral anterior chamber with the thickness of the cornea. This measure

can be derived when a narrow slit of light shines within the limbus, i.e. the edge between the cornea and the sclera. As a consequence, within the entire set of images acquired during the scan, only a few of these (referred to as *central* images) can be used to measure the ACA. Both previous and subsequent acquisitions, where the light is placed respectively on the left and on the right of the limbus, must be discarded. Indeed, since the patient eye position may change between different measurements, an a-priori images selection can not be performed. Consequently, a wider area must be scanned.

The outcome of a single measurement, that is, the dataset coming from the acquisition system consists of two scans of the entire eye, each one composed of 60 images with a $1600x1200$ px resolution. As an example, a set of images acquired during a scan is represented in Figure 6.2.

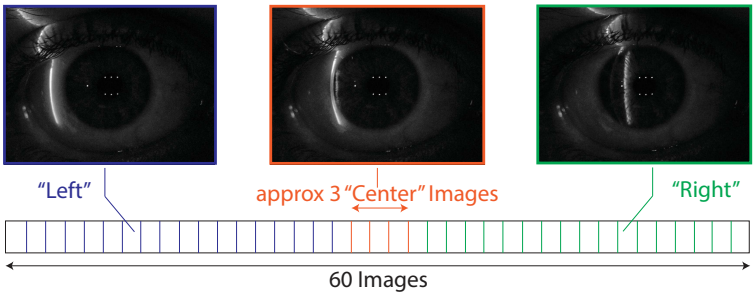

Figure 6.2: Example of a set of images acquired during a scan.

The approach followed in this work is based on a CNN, solving a three-class image classification problem. The CNN takes as inputs the images acquired by the setup and performs the classification activity by tagging the images as *left*, *central* or *right* respectively labeled with 0, 1, 2.

In the following, after a comparative analysis among different ML approaches, two CNNs that revealed most suitable to solve the classification problem, will be identified. Both of them have been then specifically analyzed, through an objective approach presented in Section 6.5. As a last remark, it is worth observing that there are no other contributions in the literature addressing an automatic procedure to perform the VH maneuver, whereas other works focus on the of ML techniques to process

images acquired with the standard manual VH technique.

6.3 ML–based classification Technique

In the following, the NN design is specifically addressed. At first, the best ML techniques for the considered application has been chosen, by exploiting a comparative approach. An investigation of the adopted CNNs is then proposed. Several networks structures have been trained, since the level of complexity of the given problem does not preclude the selection of any specific model, nor imposes an *a priori* choice of one model with respect to another. Therefore, classification performance has been tested and compared, by using the metrics introduced in Section 2.10, of some ML algorithms. In particular, SVM, K-Means and CNN structures have been compared. The latter have been chosen, with the respect to non-convolutional NN, as they generally shown a better behavior when dealing with image classification problems. In general, it would be important to obtain high values of both recall and precision, but for the application considered in this work, the precision is more important. Indeed, if some *left* or *right* images are wrongly interpreted as central, the measurement outcome taken from those images becomes intrinsically less accurate. Results are shown in Table 6.1.The performance of the different machine learning approaches over a *test data-set*, composed by 8677 non-augmented images, have been assessed in terms of validation accuracy (VAL-ACC), *center* class prediction recall (CTR-REC) and precision (CTR-PR), single frame prediction time (SFPD) and the respective standard deviation over the whole test data set. It is worth observing that, the *test data-set* used for the comparison has not been used before during the training phase. Moreover, the total number of parameters of each CNN, regarding convolutional layers only, have been reported to give an idea of overall networks complexity.

Among the different ML techniques, CNNs show a superior performance in terms of accuracy if compared to both the linear SVM classifier and the unsupervised k-means approaches. It is worth noting that the SVM accuracy is quite high while prediction of *center* images is not equally acceptable. Indeed, the linear classifier can correctly predict most of the *left* or *right* images, while confusing the central images. The same

Table 6.1: Classification Method Comparison

Method	VAL-ACC	*CTR-PR*	*CTR-REC*	SFPD (ms)	σ SFPD (ms)	Parameters
AlexNet	98.82 %	86.04 %	86.04 %	29.98	3.43	6088768
VGG16	99.02 %	88.44 %	87.18 %	33.42	3.53	14713536
ResNet50	98.29 %	79.56 %	82.05 %	39.94	6.28	23581440
SVM	92.90 %	26.92 %	7.98 %	0.39	0.21	-
K-means	59.31 %	3.73 %	12.82 %	0.71	0.25	-

can be said for the k-means ML algorithm, but with even worst performances: in this case, the unsupervised method failed to recognize a common pattern among the presented training images, hence resulting in a significantly lower performance in the validation phase.

From Table 6.1, AlexNet and VGG16 [155] are the networks providing the best performance in terms of precision and recall, and in general with respect to all the paramenters. In the table are reported also the mean and the standard deviation of the so-called SFPD. As can be seen, CNNs present higher values of SFPD. From Table 6.1 Alexnet revealed a lower complexity in terms of total parameters (last column of Table 6.1) and showed to be slightly faster than VGG16, that is however able to provide better values of precision and recall. For these reasons, both will be considered in the following analysis, and as potential candidates for the implementation within the proposed VBMs.

6.3.1 Convolutional Neural Network Design

As described in the previous section, both AlexNet [45] and VGG-16 [155] structures can be finally chosen for this application. Different Alexnet and VGG-16 settings have been tested during an extensive experimental campaign, where a trial and error methodology has been adopted. In particular, several typical parameters, such as precision and recall, have been used to optimize the training activity. Afterwards, the best training configuration for this application has been chosen and it is presented in this section. Moreover, as the adopted method is a supervised ML technique, I took particular care of the labelling process. The collected images have been carefully divided into the three classes, by exploiting the guidelines of expert ophthalmologists. *Relu*

has been adopted, both for AlexNet and VGG-16, as it is a commonly used activation function [156], that is in charge to manage the Input-Output behavior of neurons, as already discussed in Section 2.5. Moreover, in this situation, it achieves excellent performances and it demonstrated to learn faster. A slight modification has been made in respect to the typical AlexNet structure: a bilinear interpolation algorithm has been exploited to obtain resized $400x300$ px input images. It is worth noting that downsized images are used only for classification purposes, while the ACA measurement will be performed with native resolution ones. Indeed, there are no network performance improvements when using the typical $227x227$ px AlexNet input size. Likely, an higher quality of the image is more important than a fine tuning of layers and kernel sizes. Figure 6.3 shows the employed network structure.

As it is possible to see from Figure 6.3, a Dropout Rate has been set to 50%.

The VGG-16 architecture, is basically formed by the same components already discussed for the AlexNet one. Actually, the difference resides in the number and dimensions of the various layers.

Data preparation is a fundamental task to be done before the training stage. Hundreds of acquisitions were made and the data have been manually split into the three classes to train the network. Data are normalized between 0 and 1, and then 70% of the total have been used to train the network, while the other portion for validation purposes. One important feature of the available dataset is the low number of *central* images (i.e. those ones to be identified with the network), usually three or four out of 120 images acquired during the measurement procedure. To over the hump, *data augmentation* techniques have been adopted. Data augmentation usually foresees the generation of modified training examples, applying different and pre–tuned image transformations, e.g. horizontal and vertical shifts, rotations and brightness modification, only to name a few. This technique is typically used to generalize the behavior of the network, trying to generate new and meaningful examples targeted for the application. Instead, data augmentation has been used also to generate more *central* images, this way giving the possibility to train the network with comparable amounts of *left*, *right* and *central* images. The data augmentation technique has been applied to the already labelled images, to ensure that the augmented images are correctly labelled.

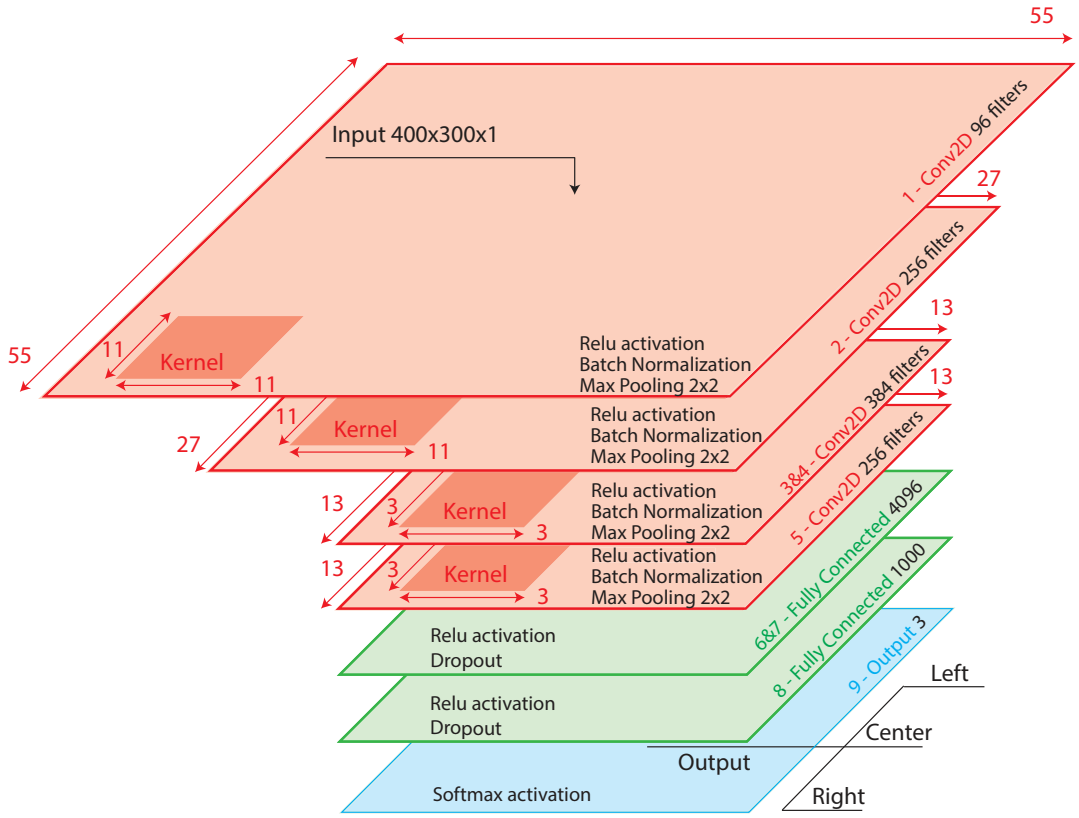

Figure 6.3: AlexNet [45] structure.

In Table 6.2 are listed the chosen data augmentation parameters.

Table 6.2: Data Augmentation Setup Parameters.

Horizontal Shift Range (% of the width)	$(-1\%, 1\%)$
Vertical Shift Range (% of the height)	$(-1\%, 1\%)$
Rotation Range (deg)	$(-7, 7)$
Brightness Multiplication Factor Range	$(0.5 - 1.75)$
Zoom Range (% of the picture size)	$(90\% - 110\%)$

Indeed, little movements to the right and to the left, rotations and zooms may occur. Furthermore, different brightness levels could take place in different surrounding environments. After the data preparation stage, the prepared data set had the properties listed in Table 6.3

Table 6.3: Train Data Set Properties.

	Original	Original Training	Augmented	Total Training
Left	8537	5976	5711	11637
Central	593	416	14994	15410
Right	7877	5513	5471	10984

In Table 6.3, as said before, the original images have been splitted in validation and training data, being respectively the 30% and 70% of the total. Afterwards, only the training images have been augmented.

6.4 Neural Network Evaluation

The CNN *AlexNet* and *VGG-16* described in Section 6.3.1 have been implemented in Python, within the *Keras* (Tensorflow Version 2.1.0) framework and trained according to the set–up described in Table 6.4. Clearly, as the hyper-parameters have been chosen after a trial-and-error stage, the final settings can be slightly different for each CNN.

At each training stage, the error gradient was calculated through the Stochastic

Table 6.4: CNN training settings.

	AlexNet	**VGG-16**
Optimizer	SGD	Adam
Learning Rate (LR)	10^{-2}	$2 \cdot 10^{-4}$
Epochs	75	150
Batch Number (BN)	128	332
Loss Function	SCC	SCC
Validation Data (% of original images)	30	30

Gradient Descent (SGD) algorithm for Alexnet, and by exploiting the Adam optimizer [157] for the VGG-16 approach. By controlling the error gradient the optimizer updated the weights so that the error decreased step by step. The next weights choice was done by evaluating the *error cost*, through a specific loss function. Given the training settings of Table 6.4, considering the n–th observation was labeled as a specific class c with probability $p_{n,c}$ and N was the total number of observations, the Sparse Categorical Crossentropy (SCC) loss function was expressed as:

$$Loss(p_{n,c}) = -\frac{1}{N}\sum_{n=1}^{N}[log(p_{n,c})]. \tag{6.1}$$

The resulting accuracy curves for train and test data (as an example, only for the Alexnet structure) are shown in Figure 6.4.

It is worth noting that in this test case the train dataset was used several times to feed the network, 75 and 150 Epochs for AlexNet and VGG-16, respectively. The training has been stopped when the loss and accuracy did not improve for several consecutive epochs, to avoid overfitting. Results are encouraging since it is possible to achieve high accuracyvalues, already presented in Table 6.1. Despite this, a more accurate analysis is needed, and a deeper evaluation is now conducted by means of different metrics. Indeed, the objective of the network is to identify images belonging to the *center* class, that has very low validation examples. For this reason, high accuracy values can be obtained also if the *central* class is not well predicted. Moreover, there is an impact of the threshold value on the precision. The threshold is defined as the

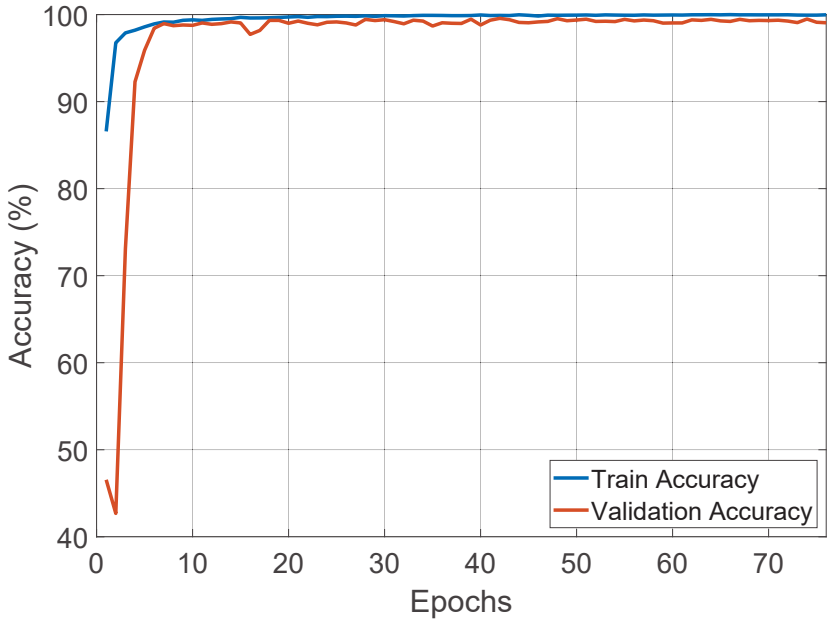

Figure 6.4: Training and Validation Accuracy improvement during the training phase.

probability level above which an observation is labeled as *center*. The resulting recall and precision curves obtained are shown in Figure 6.5 and Figure 6.6 for AlexNet and VGG-16, respectively.

An evident effect is that, as the threshold increases, recall decreases with increasing precision. Indeed, a threshold increase reflects on a lower number of total positive, i.e. TotP = TP + FP. The growth of the precision is a clear indication that the total positive decrease goes together with a decrease of false positives that mostly become true negatives. The recall decrease is slighter than the precision increase. This allows choosing a high value of the threshold to increase the precision of the whole network. It is worth observing that all the reasoning is made on the global number of test sequences, but high values of the threshold may involve in a small number of total positive for a single data–set. For this reason, in this work the chosen threshold is 89%, allowing to achieve a precision much greater than the 80% for both the CNNs.

The trained network can be finally used to predict *central* images on the newly

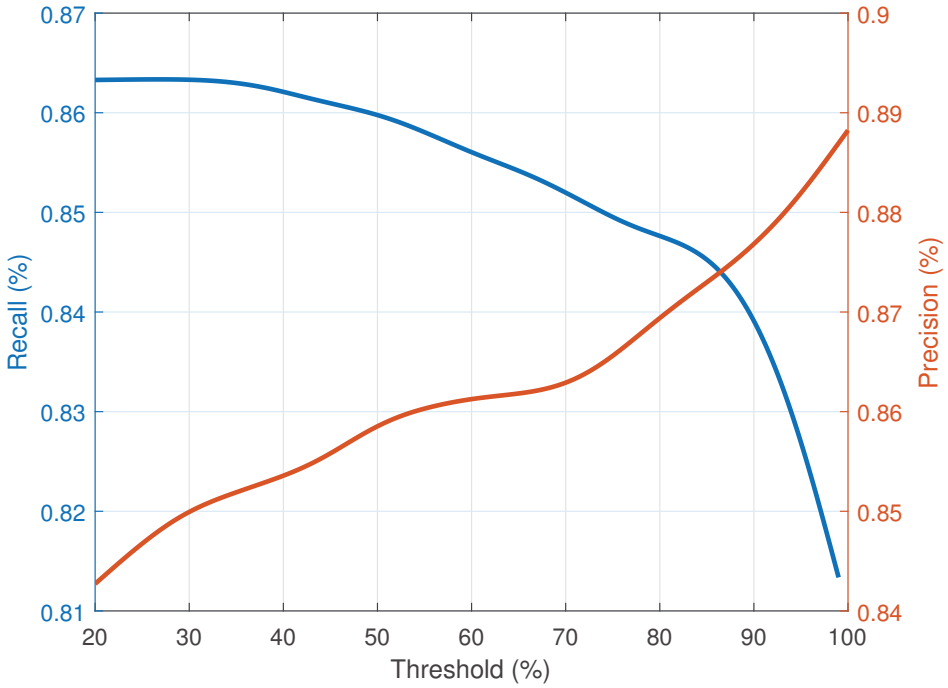

Figure 6.5: Precision and Recall: AlexNet

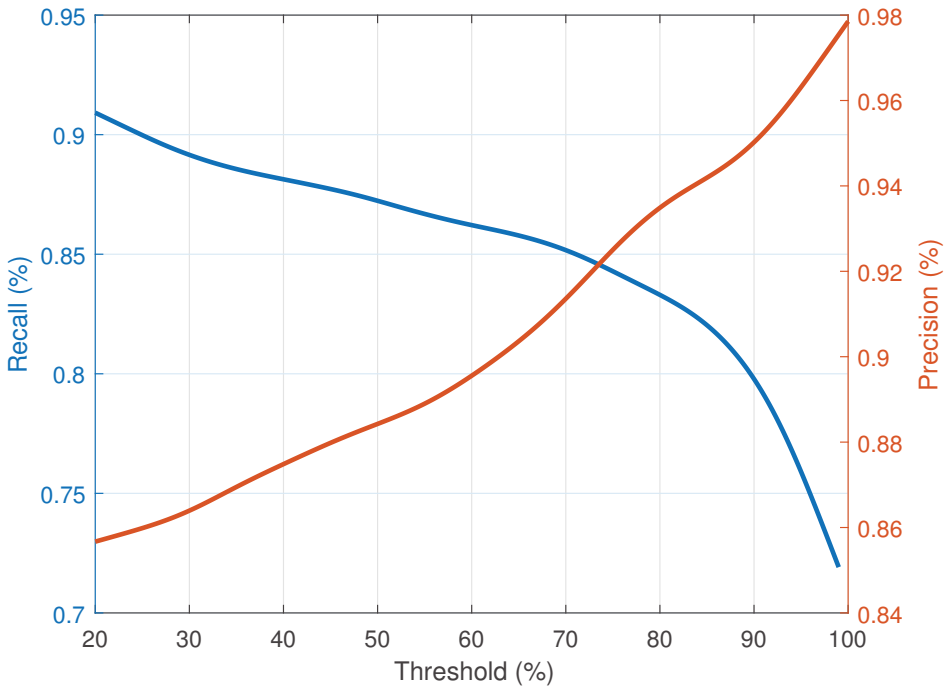

Figure 6.6: Precision and Recall: VGG-16

acquired data set. The classification algorithm can be run in the described embedded system, with additional few seconds of execution that does not increase the acquisition time, as it is performed after the scans are completed. This is surely an important aspect to underline, as it is demonstrated that patient's ability to look steady toward the fixation target, is time-limited.

6.5 Test results

As already stated, one of the big challenges on the usage of ML technique in the IoM panorama, is the need for a metrological evaluation. In this section, the goodness of the CNNs performance is evaluated, from a metrological point of view, through the study of the difference in pixels between the position of the limbus and the position of the light line on the eye surface: the lower this difference the higher the CNN accuracy. This procedure has been divided into 4 steps: i) recognition of the iris from a reference image to have a reference for the intensity profile extraction; ii) extraction of the limbus position from a reference image captured at the first instant of the measurement procedure; iii) evaluation of the line position for every image labeled as *central*; iv) computation of the difference between these two positions. It is worth observing that all the procedure is conducted in a semi-automatic way, thus revealing to be not directly applicable to identify the center images. Indeed, the aim is to build a totally automatic VBMs system. Aiming at a metrological evaluation of the NN behavior, the algorithm has been tested with a new dataset, that was not used for training, neither for previous validation purposes. The dataset was collected by expert ophthalmologists at IRCCS Fondazione G.B.Bietti. The dataset consist of a total of 36 different subjects by expert ophthalmologists in a clinical environment. As before, the number of *center* images chosen by the network ranged from 3 to 5 per each measurement. The total amount of analyzed central images was 140. The dataset includes eyes of four different colors: brown, blue, black, and green. The number of measurements comprised in this new dataset for each color is presented in Table 6.5.

Results are summarized in Table 6.6.

It is worth observing that the mean error is always positive. Indeed, by suitably

Table 6.5: New dataset properties.

Color	Number of Measurements
Black	14
Brown	13
Green	11
Blue	12

Table 6.6: Mean and standard deviation of the error E between the limbus and the light line position for different iris pigmentation.

	AlexNet		VGG-16	
Color	μ (px)	σ (px)	μ (px)	σ (px)
Black	1.09	4.36	$2,6$	$3,2$
Brown	1.20	4.28	$3,7$	$5,9$
Green	6.69	5.19	$6,2$	$5,6$
Blue	7.51	5.48	$7,2$	$6,2$
Total	4.12	5.67	$5,37$	$6,1$

labeling the training data, the network has been instructed to choose central images which present the line slightly inside the iris, rather than slightly outside. Actually, when the line is somewhat inside the iris, it is still possible to see the refraction of the light, and derive the VH grade. Results depend on the specific hardware used (in particular the light line thickness) which makes substantially this error totally negligible, as the thickness of the light line is always greater than the obtained error.

Conclusions

In this Ph.D. thesis the novel concept of smart and distributed measurement system has been deeply analyzed. In the first chapters, the key enabling technologies (IoT, AI and ML and wireless communication protocols) have been deeply analyzed. In particular, from the derived literature statistics it was possible to underline that the application of IoT systems to derive measurements (for example in the field level of the novel smart factory) is gaining much research interest during the last years. Moreover, from an extensive analysis of the literature, the requirements for these novel smart systems have been introduced and analyzed. Between others, the most important ones are surely related to timeliness and real-time behavior of the system. We demonstrated that these requirements have an impact on the measurement accuracy of the sensor network. In particular, following the GUM, we have derived the relationship between measurement uncertainty and uncertainty on the knowledge of the delay introduced by the communication network. In substance, the measurement uncertainty is proportionally dependent from the uncertainty on the knowledge of the delay. This observation underlines, again, the need for deterministic systems.

Being the communication delay such an important parameter, one of the big challenges, in this context, revealed to be the specific communication protocol adopted, especially if wireless communication is needed. For this reason, widespread used wireless communication strategies have been deeply analyzed, and the novel TSN standardization activity has been introduced. The latter, being specifically designed for time critical systems, is surely attractive in the field.

The experimental part of the thesis started with an analysis of RA strategies for

Wi-Fi networks, as they proved to increase the performances of IEEE 802.11 networks in terms of network delay. Despite this, an analysis of the impact of the RAA has been investigated. The presented experimental results proved that the impact on the round trip time is limited, and that algorithms that take more time to execute are more efficient from a RA point of view, thus proving to be efficient in the field. Despite this, different RA strategies have different execution times, so an analysis of this aspect before choosing the RAA is needed. As a matter of fact, also if not included in this thesis for sake of brevity, we investigated a RL RA strategy for Wi-Fi in a literature work [158], that proved to be efficient. Moreover, a RL strategy is interesting from a computational point of view, as it foresees only to choose the rate from a Look-Up Table.

Subsequently, LoRa networks have been addressed. A meaningful test case has been presented; from the latter it was possible to underline that also LoRa networks, characterized by long communication ranges, low power consumption and low data rates can be used in IoM. Moreover, it was possible to underline which problems of LoRa technologies need to be overcame, such as an optimization of the trade-off between packets correctly received and energy consumption and the DC limitation. The former has been solved by using an RL based parameter adaptation strategy. Subsequently, the DC limitations have been solved by introducing hybrid channel access strategies.

As a matter of fact, the proposed techniques underline that both Wi-Fi and LoRa can be applied in the IoM scenario, after a careful optimization of the protocols. In particular, Wi-Fi together with 5G, can be applied when high data rates are needed. Some simulations on 5G are presented in Appendix A, that revealed to introduce very low latency (in the order of 10 ms). I contextualized 5G in a meaningful IoM application.

As a secondary project, I also tried some ML based post-processing techniques. These techniques are surely important in the IoM scenario, but a precise characterization of their impact in the measurement accuracy is needed.

As a matter of fact the IoM context exploits a plethora of different and interdis-

ciplinary techniques. Indeed, IoT systems (also the ones used to smartly derive measurements) are complex systems, that still pose several different research challenges, that this work helps to underline. Wireless communication and Artificial Intelligence techniques are, in facts, surely between them, together with a metrological characterization of IoT systems. In particular, this work underlined that these technologies, that deeply modified our everyday life, must be optimized to cope with the stringent requirements coming from the IoM. Also the aforementioned IoM concept (i.e. the design and analysis of smart and distributed measurement systems) is somehow new. It was common practice to refer to the industrial measurement systems for example, but the reality is more complex, and the same requirements can be generalized. The same technologies, analysis, methodologies can be applied into different fields where accurate measurements are needed. In Appendix A, for example, I will present a totally different application field where the same concept can be applied. For this reason, this work is deliberately a cross-party thesis, comprising the analysis of different technologies and applications. I hope to have been part of the process that will bring to the development of complete, accurate and innovative systems devoted to measurements.

Appendix A

IEC 61850 over 5G

In this appendix, the results of some simulations carried out in a 5G network assessment are presented. These last results have been added in order to demonstrate the interdisciplinary nature of this thesis, and in general the IoM context. Indeed, as already stated from the first chapters of this work, the IoM paradigm can be applied to different application fields. At the end, it is possible to see from this chapter that the already discussed requirements are the same also for this specific test-case. These results have been added in appendix only because can be considered preliminary results and further analysis is needed. These results were obtained in order to test the suitability of 5G networks in the IEC 61850 context, that is briefly presented in Section A.1. After the analysis of the simulated scenario, some results on the latency introduced by the 5G network are presented in Section A.3.

A.1 The IEC 61850 context

Modern power systems are experiencing a rapid transition from the traditional monitoring and control paradigm, mainly based on analog quantities, to a novel framework where both measured quantities and control messages are conveyed in a digital format [159], [160]. In this context, Distributed Measurement Systems are becoming increasingly important. The sensing infrastructure has being deployed in the most sensitive

node of the grids, particularly the electrical substations, where different kinds of instruments allow for monitoring in real-time the main quantities of interest [161]. In this context, Phasor Measurement Units (PMU)s represent one of the most promising solutions as they produce time-stamped measurements of voltage and current phasor. IEC/IEEE 60255-118-1 (PMU-Std) [162] is the reference standard in the field, and defines the performance requirements in terms of measurement accuracy, latency, response time. To this regards, it is important to underline that this application becomes a good example of non-industrial IoM system, with the same requirements already discussed in the previous parts of this work. The aforementioned PMU-Std introduces also some synchronization constraints: as already stated, synchronization between devices is an undeniable requirement in time critical systems. Based on the associated time-stamps, it is possible to aggregate and compare PMU measurements coming from different nodes and perform other crucial analysis.

Another important standard is the IEC 61850 (IEC-Std) [163], that defines the substation communication protocols and the need for interoperability between systems from different vendors [164]–[166]. However, it is worth noticing that many communications within a digital substations are often time critical and fully cabled solutions are not always feasible due to implementation or economical reasons [167], [168]. Based on these considerations, the recent literature has been discussing the practicality and potential benefits of 5G mobile communication technology in power systems, particularly in distribution grids where the distances between substations are expected to be limited and a single or few antennas may be sufficient to cover the entire range of interest [169], [170].

The IEC-Std has gained an ever-increasing research interest in the last years. In a more quantitative approach, I carried out an extensive literature research on Scopus, by using 5 different search parameters, applied on the keywords, abstract and title fields. The total number of articles found is reported in Table A.1 and a per–year statistic is reported in Figure A.1.

In this regard, it is interesting to observe how the highest number of research contributions corresponds to the publication of the IEC Technical Report 61850-1 (IEC-TR) [171] in 2013 that defined the communication between intelligent electronic

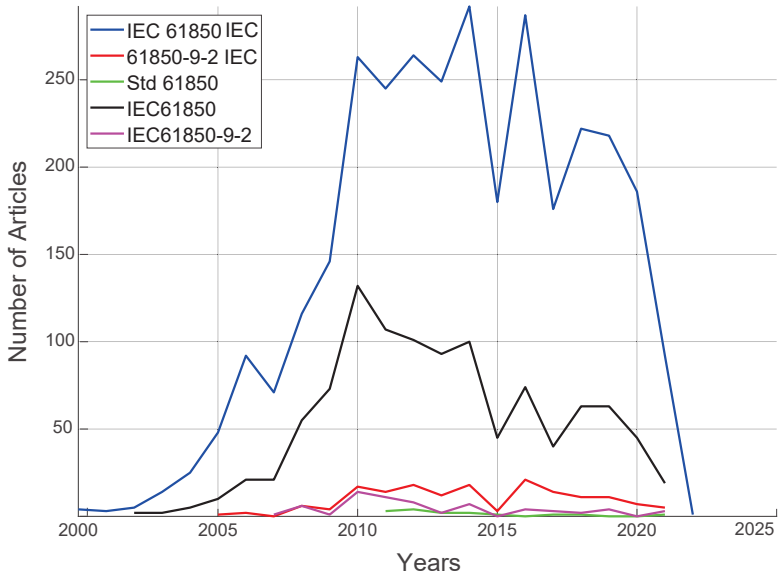

Figure A.1: Scopus Literature Review: per - year number of articles per topic.

devices in such a system, and the related system requirements. Indeed, the IEC-TR was the first attempt to extended the scope of the IEC-Std communication protocol and to make it compatible with power quality and distributed monitoring and control applications.

Despite the relevant number of publication, though, it is worth noticing a significant lack of rigorous performance assessment and feasibility analysis in the I&M community,

Table A.1: Scopus Literature Review: Number of Articles per Topic

Search Parameter	**Total number of Articles**
IEC 61850	3199
IEC 61850	1071
IEC Std 61850	15
IEC 61850-9-2	164
IEC 61850-9-2	66
IEC Std 61850-9-2	5

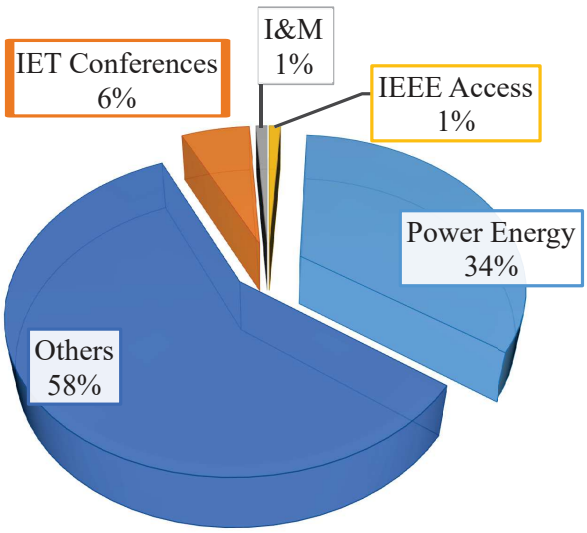

Figure A.2: IEC 61850 Keywords: association to IEEE communities.

as testified by Figure A.2. Indeed, it is interesting to observe how only 1 % of the listed research contributions address the topic from a measurement science or even metrological perspective. In order to fill this gap, this Chapter focuses on the impact of communication delay, that is still a metrological issue. In fact, as underlined in Section 1.2 the transmission delay has an impact on the measurement uncertainty.

A.2 Simulation Setup

For this analysis, a real-world power grid is considered, namely a 10-kV three-phase distribution network, located in the Netherlands and operated by the DSO Alliander. The network topology as well as its main parameters are thoroughly described in [172].

W assume to populate each node with a PMU operating with a reporting rate of 50 fps and transmitting its measurements to the Phasor Data Concentrator located in node 1. The behaviour of the 5G communication infrastructure has been properly analysed by means of a suitable simulation assessment, developed exploiting the popular OMNeT++ simulator [173]. OMNeT++ has been extensively used to simulate com-

munication networks in the last years, and it is a C++ discrete event simulator. It is worth observing that, lately, OMNeT++ introduced several new communication technologies, comprising for example TSN and 5G. The latter is of particular importance for this study, and it has been introduced in OMNeT++ thanks to the developers of the Simu5G project.

The Simu5G simulator [174] is an OMNeT++ - based simulator, specifically designed for 5G networks. This simulator comprises suitable modelizations of both 3GPP-compliant protocol stack and the transmission channel [175]. The basic structure of a 5G network is depicted in Figure A.3.

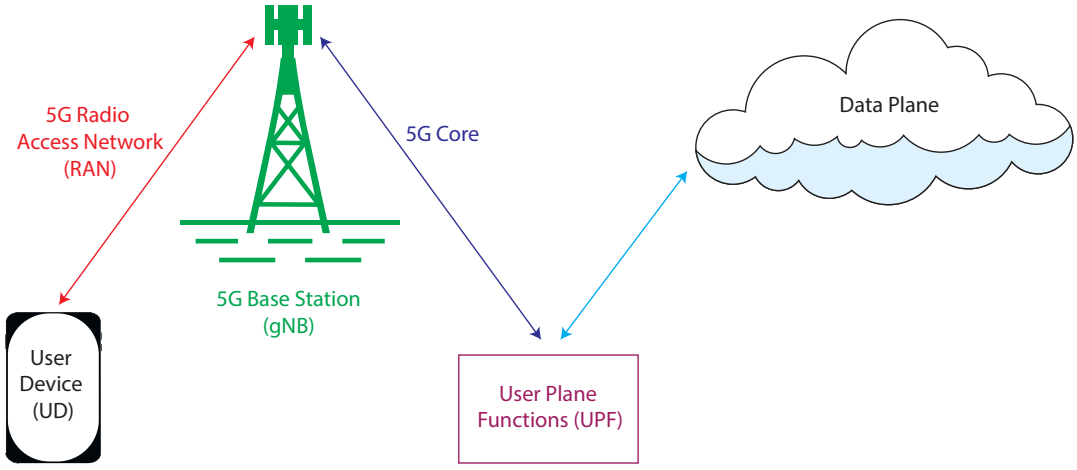

Figure A.3: 5G Architecture.

In particular, the 5G network is managed by a base station (gNodeB, gNB), that can be considered an evolution of the older 4G ones (eNodeB, eNB). It is worth observing that the simulator offers total compatibility with the 4G network, allowing to simulate mixed 4G/5G scenarioes. As a matter of fact, each User Device (UD) can forward messages to the nearest gNB, through the so-called Radio Access Network (RAN) where data communication occurs at Open System Interconnection (OSI) layer 2. The used (and also developed in the considered simulator) protocols, are listed in Table A.2.

Table A.2: 3GPP modeled protocols for each OSI reference layer.

OSI Layer	Protocols
Data Link Layer	PDCP RLS MAC
Physical	PHY

The simu5G simulator is a very interesting and powerful tool, that allows to accurately analyse 5G networks. Simu5G gives also the possibility to perform emulation activities [176], [177]. As a matter of fact, it is really important to underline that the simulator used in this study has been properly validated and, with the respect to other popular 5G simulators like the Matlab-based Vienna 5G SL [178], it allows to simulate the whole stack and not only the lower layers.

The considered simulated network is depicted in Figure A.4.

As a matter of fact, the network in [172] has been reproduced by modelling the substations as UD, in charge of collecting measures and sending them through a network of base stations. The 5G base stations have been chosen pointing to the full coverage of the area, according to a maximum transmission range of 500m. From Figure A.4 it is possible to recognise all the aforementioned parts of a 5G architecture: UDs, gNBs, User Plane Functions (UPF)s and Data Planes, represented by the servers. In the next section, some preliminary results, using a 5G ULL network, are discussed.

A.3 Latency Results

A 5G ULL network have been deployed, according to what already said in the previous section. Simulations have been conducted by using the parameters listed in Table A.3.

The 5G It is worth observing that, according to what said before, measurements are generated periodically, and are all synchronized between stations. In particular, I evaluated two different setups. Firstly, I set up the communication such as all the

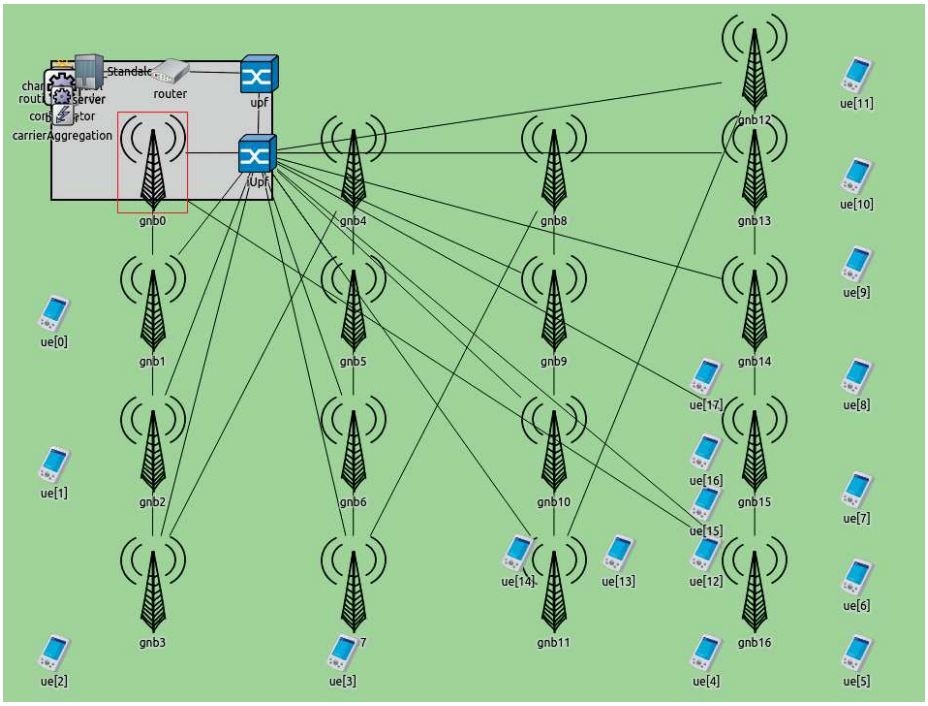

Figure A.4: Simulated network.

data are sent to the first substation (UE[0]) in Figure A.4. In the following, I will refer to such configuration as D2D communication. Secondly, I set up a network where all the stations send data to the server, namely UpLink (UL) configuration. Then, it is possible to imagine that the server is placed exactly nearby UE[0]. The Cumulative distribution function of the communication delay is depicted in Figure A.5.

The results in Figure A.5 are totally reasonable, as in order to perform D2D both an UL to the server and DownLink (DL) phase from the server are needed. If the server can be placed next to the first substation the UL strategy outperforms D2D. Other useful statistics, for the chosen UL scenario, are reported in Table A.4.

Where the DER is calculated as per Eq. (1.2). As a matter of fact, the results are encouraging. On one side, they present a significant reduction of the communication delay if compared to the previously published results, related to a 4G communication infrastructure. On the other side, the overall delay caused by the communication in-

Table A.3: Parameters used for the simulations.

Parameter	Value
Packet Size	34 (bytes)
Simulation Time	1 (h)
Packet Generation Frequency	50($\frac{pack}{s}$)
Transmission Tipology	5G ULL

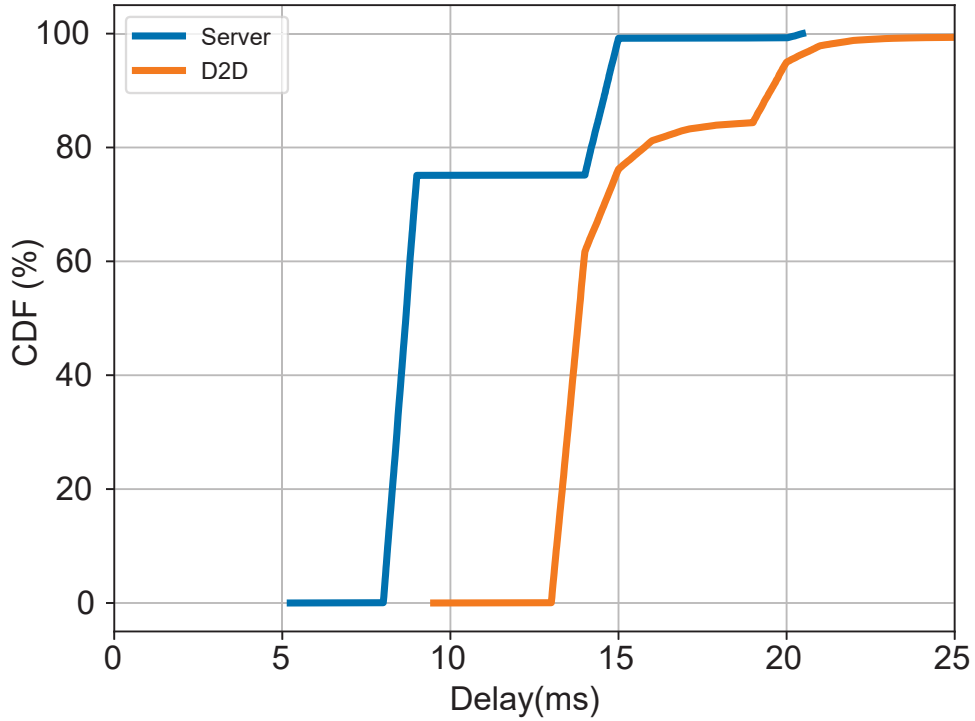

Figure A.5: Cumulative Distribution Function of the communication delay, both D2D and UL.

frastructure and the limited amount of lost packets are perfectly in line with most monitoring and control applications. In this regard, it is important to observe that state estimation applications have update rates that depend on the slowest measure-

Table A.4: Results of the simulations.

Parameter	Value
Communication Delay: Mean Value	10.03 (ms)
Communication Delay: Standard Deviation	2.73 (ms)
DER	99.94 (%)

ment data stream included in the system, and they usually overcome the tens of ms. Similarly, in control applications as under-frequency load shedding there exist waiting and settling times in the order of hundreds of ms to let the system settle down after each control action.

Based on these considerations, it is reasonable to say that the usage of 5G seems convenient as it allows to handle dense networks, with a lot of devices, and points to a very speed communication, giving the possibility to lower down the communication delays until 1 to 4 ms. Current installations achieve slightly higher transmission times, like the ones I experienced in this study.

Bibliography

[1] Guo-Jian Cheng, Li-Ting Liu, Xin-Jian Qiang, et al. "Industry 4.0 Development and Application of Intelligent Manufacturing". In: *2016 International Conference on Information System and Artificial Intelligence (ISAI)*. 2016, pp. 407–410. DOI: 10.1109/ISAI.2016.0092.

[2] Jian Qin, Ying Liu, and Roger Grosvenor. "A Categorical Framework of Manufacturing for Industry 4.0 and Beyond". In: *Procedia CIRP* 52 (2016). The Sixth International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV2016), pp. 173–178. ISSN: 2212-8271. DOI: https://doi.org/10.1016/j.procir.2016.08.005.

[3] Boon-Yaik Ooi and Shervin Shirmohammadi. "The potential of IoT for instrumentation and measurement". In: *IEEE Instrumentation & Measurement Magazine* 23.3 (2020), pp. 21–26. DOI: 10.1109/MIM.2020.9082794.

[4] K. Ashton. "That 'Internet of Things' Thing". In: *RFID journal* (June 2009).

[5] Lei Shu and Zhang Zijiang. "Exploration and thoughts on environmental monitoring systems for the preservation of cultural relics based on LoRa and NB - IoT —taking Sichuan Museum as an example". In: *Sciences of Conservation and Archaeology* 34.4 (2022), pp. 114–122. DOI: 10.16334/j.enki.en31-1652/k.20210302067.

[6] Iran Carlos Caria Sacramento, Emerson de Andrade Marques Ferreira, Vivian de Oliveira Fernandes, et al. "Low-cost multipurpose sensor network integrated

with iot and webgis for fire safety concerns". In: *Acta Scientiarum - Technology* 45 (2023). DOI: 10.4025/actascitechnol.v45i1.61451.

[7] Noel Bristow, Saravanan Rengaraj, David R. Chadwick, et al. "Development of a LoRaWAN IoT Node with Ion-Selective Electrode Soil Nitrate Sensors for Precision Agriculture". In: *Sensors* 22.23 (2022). DOI: 10.3390/s22239100.

[8] Md Whaiduzzaman, Alistair Barros, Moumita Chanda, et al. "A Review of Emerging Technologies for IoT-Based Smart Cities". In: *Sensors* 22.23 (2022). DOI: 10.3390/s22239271.

[9] Mohamed A. Ahmed, Sebastian A. Chavez, Ali M. Eltamaly, et al. "Toward an Intelligent Campus: IoT Platform for Remote Monitoring and Control of Smart Buildings". In: *Sensors* 22.23 (2022). DOI: 10.3390/s22239045.

[10] Bo Yang, Jun Tang, Xiaofeng Dong, et al. "Power Inspection Design by Internet of Things and RFID Technology in Smart City". In: *Microprocessors and Microsystems* 96 (2023). DOI: 10.1016/j.micpro.2022.104510.

[11] Shadab Faham, Abdollah Salimi, and Raouf Ghavami. "Electrochemical-based remote biomarker monitoring: Toward Internet of Wearable Things in telemedicine". In: *Talanta* 253 (2023). DOI: 10.1016/j.talanta.2022.123892.

[12] Pravin Kulurkar, Chandra kumar Dixit, V.C. Bharathi, et al. "AI based elderly fall prediction system using wearable sensors: A smart home-care technology with IOT". In: *Measurement: Sensors* 25 (2023). DOI: 10.1016/j.measen.2022.100614.

[13] Waleed Bin Qaim, Aleksandr Ometov, Antonella Molinaro, et al. "Towards Energy Efficiency in the Internet of Wearable Things: A Systematic Review". In: *IEEE Access* 8 (2020), pp. 175412–175435. DOI: 10.1109/ACCESS.2020.3025270.

[14] Fangchun Yang, Shangguang Wang, Jinglin Li, et al. "An overview of Internet of Vehicles". In: *China Communications* 11.10 (2014), pp. 1–15. DOI: 10.1109/CC.2014.6969789.

[15] Keyur K Patel, Sunil M Patel, and P Scholar. "Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges". In: *International journal of engineering science and computing* 6.5 (2016).

[16] Eduardo S.F. dos Santos, William B. Xavier, Ricardo N. Rodrigues, et al. "Vision Based Measurement applied to Industrial Instrumentation". In: *IFAC-PapersOnLine* 50.1 (2017). 20th IFAC World Congress, pp. 788–793. ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2017.08.509.

[17] Gobert Lee and Hiroshi Fujita. *Deep learning in medical image analysis: challenges and applications*. Vol. 1213. Springer, 2020.

[18] J. Zhong, Z. Liu, Z. Han, et al. "A CNN-Based Defect Inspection Method for Catenary Split Pins in High-Speed Railway". In: *IEEE Transactions on Instrumentation and Measurement* 68.8 (Aug. 2019), pp. 2849–2860. DOI: 10.1109/TIM.2018.2871353.

[19] Marco Venturelli, Guido Borghi, Roberto Vezzani, et al. "Deep Head Pose Estimation from Depth Data for In-Car Automotive Applications". en. In: *Understanding Human Activities Through 3D Sensors*. Ed. by Hazem Wannous, Pietro Pala, Mohamed Daoudi, et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 74–85. ISBN: 978-3-319-91863-1. DOI: 10.1007/978-3-319-91863-1_6.

[20] Yan Tong, Wei Lu, Yue Yu, et al. "Application of machine learning in ophthalmic imaging modalities". In: *Eye and Vision* 7 (2020). DOI: 10.1186/s40662-020-00183-6.

[21] Md Zakirul Alam Bhuiyan, Sy-Yen Kuo, and Guojun Wang. "Guest Editorial: Trustworthiness of AI/ML/DL Approaches in Industrial Internet of Things and Applications". In: *IEEE Transactions on Industrial Informatics* 19.1 (2023), pp. 969–972. DOI: 10.1109/TII.2022.3201588.

[22] S. Vitturi, C. Zunino, and T. Sauter. "Industrial Communication Systems and Their Future Challenges: Next-Generation Ethernet, IIoT, and 5G". In: *Proceedings of the IEEE* 107.6 (June 2019), pp. 944–961.

[23] M. Wollschlaeger, T. Sauter, and J. Jasperneite. "The Future of Industrial Communication: Automation Networks in the Era of the Internet of Things and Industry 4.0". In: *IEEE Industrial Electronics Magazine* 11.1 (2017), pp. 17–27. ISSN: 1932-4529. DOI: 10.1109/MIE.2017.2649104.

[24] Francesco Branz, Riccardo Antonello, Matthias Pezzutto, et al. "Drive-by-Wi-Fi: Model-Based Control Over Wireless at 1 kHz". In: *IEEE Transactions on Control Systems Technology* (2021), pp. 1–12. DOI: 10.1109/TCST.2021.3094865.

[25] *ISO Guide to the Expression of Uncertainty in Measurement (GUM)*. https://www.iso.org/standard/50461.html, [Online; Accessed January 4, 2023].

[26] PTB. *Metrology for the Digitalization of the Economy and Society*. https://www.ptb.de/cms/fileadmin/internet/forschung_entwicklung/digitalisierung/PTB-Digitalisierungsstudie_2018_EN.pdf, [Online; Accessed January 4, 2023].

[27] A. Willig. "Recent and Emerging Topics in Wireless Industrial Communications: a Selection". In: *IEEE Trans. on Industrial Informatics* 4.2 (May 2008), pp. 102–124.

[28] V. C. Gungor and G. P. Hancke. "Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches". In: *IEEE Transactions on Industrial Electronics* 56.10 (2009), pp. 4258–4265.

[29] Gang Zhao. "Wireless Sensor Networks for Industrial Process Monitoring and Control: A Survey". In: *Network Protocols and Algorithms* 3 (Apr. 2011). DOI: 10.5296/npa.v3i1.580.

[30] B.R. Mehta and Y.J. Reddy. *Industrial Process Automation Systems Design and Implementation*. Elsevier Inc., 2015.

[31] *Industrial Wireless Sensor Network (IWSN) Market Size, Share & Trends Analysis Report By Component (Hardware, Software, Service), By Type, By Technology, By Application, By End Use, And Segment Forecasts, 2019–2025*. Tech. rep. [Online; Accessed January 4, 2023]. Grand View Research, 2019. URL: ht

tps://www.grandviewresearch.com/industry-analysis/industrial-wireless-sensor-networks-iwsn-market.

[32] H. Xu, W. Yu, D. Griffith, et al. "A Survey on Industrial Internet of Things: A Cyber-Physical Systems Perspective". In: *IEEE Access* 6 (2018), pp. 78238–78259. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2884906.

[33] Hua-Jun Hong. "From Cloud Computing to Fog Computing: Unleash the Power of Edge and End Devices". In: *2017 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*. 2017, pp. 331–334. DOI: 10.1109/CloudCom.2017.53.

[34] Rakesh Jain and Samir Tata. "Cloud to Edge: Distributed Deployment of Process-Aware IoT Applications". In: *2017 IEEE International Conference on Edge Computing (EDGE)*. 2017, pp. 182–189. DOI: 10.1109/IEEE.EDGE.2017.32.

[35] Quoc-Viet Pham, Tuan LeAnh, Nguyen H. Tran, et al. *Decentralized Computation Offloading and Resource Allocation in Heterogeneous Networks with Mobile Edge Computing*. 2018. DOI: 10.48550/ARXIV.1803.00683. URL: https://arxiv.org/abs/1803.00683.

[36] Djabir Abdeldjalil Chekired, Lyes Khoukhi, and Hussein T. Mouftah. "Industrial IoT Data Scheduling Based on Hierarchical Fog Computing: A Key for Enabling Smart Factory". In: *IEEE Transactions on Industrial Informatics* 14.10 (2018), pp. 4590–4602. DOI: 10.1109/TII.2018.2843802.

[37] Andreas Holzinger. "Machine learning for health informatics". In: *Machine learning for health informatics*. Springer, 2016, pp. 1–24.

[38] Konstantinos G. Liakos, Patrizia Busato, Dimitrios Moshou, et al. "Machine Learning in Agriculture: A Review". In: *Sensors* 18.8 (2018). ISSN: 1424-8220. DOI: 10.3390/s18082674. URL: https://www.mdpi.com/1424-8220/18/8/2674.

[39] Fotios Zantalis, Grigorios Koulouras, Sotiris Karabetsos, et al. "A Review of Machine Learning and IoT in Smart Transportation". In: *Future Internet* 11.4 (2019). ISSN: 1999-5903. DOI: 10.3390/fi11040094. URL: https://www.mdpi.com/1999-5903/11/4/94.

[40] Massimo Bertolini, Davide Mezzogori, Mattia Neroni, et al. "Machine Learning for industrial applications: A comprehensive literature review". In: *Expert Systems with Applications* 175 (2021), p. 114820. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2021.114820. URL: https://www.sciencedirect.com/science/article/pii/S095741742100261X.

[41] A. M. Turing. "I.-Computing Machinery ans Intelligence". In: *Mind* LIX.236 (Oct. 1950), pp. 433–460. ISSN: 0026-4423. DOI: 10.1093/mind/LIX.236.433.

[42] John R. Searle. "Minds, brains, and programs". In: *Behavioral and Brain Sciences* 3.3 (1980), pp. 417–424. DOI: 10.1017/S0140525X00005756.

[43] James H. Fetzer. "What is Artificial Intelligence?" In: *Artificial Intelligence: Its Scope and Limits*. Dordrecht: Springer Netherlands, 1990, pp. 3–27. ISBN: 978-94-009-1900-6. DOI: 10.1007/978-94-009-1900-6_1. URL: https://doi.org/10.1007/978-94-009-1900-6_1.

[44] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd ed. Prentice Hall, 2010.

[45] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Neural Information Processing Systems* 25 (Jan. 2012). DOI: 10.1145/3065386.

[46] Christian Szegedy, Wei Liu, Yangqing Jia, et al. *Going Deeper with Convolutions*. 2014. arXiv: 1409.4842 [cs.CV].

[47] Kaiming He, Xiangyu Zhang, Shaoqing Ren, et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[48] Olga Russakovsky, Jia Deng, Hao Su, et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[49] Yuxi Li. *Reinforcement Learning Applications*. 2019. arXiv: 1908.06973 [cs.LG].

[50] T. Sauter. "The Three Generations of Field-Level Networks—Evolution and Compatibility Issues". In: *IEEE Transactions on Industrial Electronics* 57.11 (2010), pp. 3585–3595.

[51] M. Felser and T. Sauter. "The fieldbus war: history or short break between battles?" In: *4th IEEE International Workshop on Factory Communication Systems*. Aug. 2002, pp. 73–80. DOI: 10.1109/WFCS.2002.1159702.

[52] Max Felser. "The Fieldbus Standards: History and Structures". In: Jan. 2002.

[53] P. Danielis, J. Skodzik, V. Altmann, et al. "Survey on real-time communication via ethernet in industrial automation environments". In: *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*. 2014, pp. 1–8. DOI: 10.1109/ETFA.2014.7005074.

[54] M. Felser and T. Sauter. "Standardization of industrial Ethernet - the next battlefield?" In: *IEEE International Workshop on Factory Communication Systems, 2004. Proceedings.* Sept. 2004, pp. 413–420. DOI: 10.1109/WFCS.2004.1377762.

[55] J. Jasperneite, J. Imtiaz, M. Schumacher, et al. "A Proposal for a Generic Real-Time Ethernet System". In: *IEEE Transactions on Industrial Informatics* 5.2 (May 2009), pp. 75–85. ISSN: 1941-0050. DOI: 10.1109/TII.2009.2017259.

[56] J. Jasperneite, M. Schumacher, and K. Weber. "Limits of increasing the performance of Industrial Ethernet protocols". In: *2007 IEEE Conference on Emerging Technologies and Factory Automation (EFTA 2007)*. Sept. 2007, pp. 17–24. DOI: 10.1109/EFTA.2007.4416748.

[57] Lv Yong, Yu Hai-bin, Wang Tian-ran, et al. "Fieldbus interoperation technologies". In: *Fifth World Congress on Intelligent Control and Automation (IEEE Cat. No.04EX788)*. Vol. 4. 2004, 3620–3623 Vol.4.

[58] Fang Yanjun and Xu Jun. "An approach for interoperation between heterogeneous fieldbus systems". In: *2005 IEEE Conference on Emerging Technologies and Factory Automation*. Vol. 2. 2005, 5 pp.–243.

[59] F. Arjmandi and B. Moshiri. “Fieldbus Interoperability on Ethernet”. In: *2007 5th IEEE International Conference on Industrial Informatics*. Vol. 1. 2007, pp. 213–218.

[60] Tang Zhong, Mei Zhan, Zeng Peng, et al. “Industrial wireless communication protocol WIA-PA and its interoperation with Foundation Fieldbus”. In: *2010 International Conference On Computer Design and Applications*. Vol. 4. 2010, pp. V4-370-V4–374.

[61] T. Dang, C. Merieux, J. Pizel, et al. “On the Road to Industry 4.0: A Fieldbus Architecture to Acquire Specific Smart Instrumentation Data in Existing Industrial Plant for Predictive Maintenance”. In: *2018 IEEE 27th International Symposium on Industrial Electronics (ISIE)*. 2018, pp. 854–859.

[62] P. Bellagente, P. Ferrari, A. Flammini, et al. “Enabling PROFINET devices to work in IoT: Characterization and requirements”. In: *2016 IEEE International Instrumentation and Measurement Technology Conference Proceedings*. 2016, pp. 1–6.

[63] P. Park, S. Coleri Ergen, C. Fischione, et al. “Wireless Network Design for Control Systems: A Survey”. In: *IEEE Communications Surveys Tutorials* 20.2 (2018), pp. 978–1013.

[64] Susruth Sudhakaran, Karl Montgomery, Mohamed Kashef, et al. “Wireless Time Sensitive Networking for Industrial Collaborative Robotic Workcells”. In: (2021), pp. 91–94. DOI: 10.1109/WFCS46889.2021.9483447.

[65] Dave Cavalcanti, Stephen Bush, Marc Illouz, et al. “Wireless TSN–Definitions, Use Cases & Standards Roadmap”. In: *Avnu Alliance* (2020), pp. 1–16.

[66] J. Song, S. Han, A. Mok, et al. “WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control”. In: *2008 IEEE Real-Time and Embedded Technology and Applications Symposium*. Apr. 2008, pp. 377–386.

[67] *ISA100.11*. [Online; Accessed January 4, 2023]. URL: https://www.isa.org/isa100/.

[68] S. Hong, X. S. Hu, T. Gong, et al. "On-Line Data Link Layer Scheduling in Wireless Networked Control Systems". In: *2015 27th Euromicro Conference on Real-Time Systems*. July 2015, pp. 57–66. DOI: 10.1109/ECRTS.2015.13.

[69] Tianyu Zhang, Tao Gong, Song Han, et al. "Distributed Dynamic Packet Scheduling Framework for Handling Disturbances in Real-Time Wireless Networks". In: *RTAS* (2017).

[70] Tianyu Zhang, Tao Gong, Song Han, et al. "FD-PaS: A Fully Distributed Packet Scheduling Framework for Handling Disturbances in Real-Time Wireless Networks". In: *IEEE Computer Society* (2018).

[71] Michele Luvisotto, Federico Tramarin, and Stefano Vitturi. "A learning algorithm for rate selection in real-time wireless LANs". In: *Computer Networks* 126 (2017), pp. 114–124.

[72] F. Branz, M. Pezzutto, R. Antonello, et al. "Drive-by-Wi-Fi: testing 1 kHz control experiments over wireless". In: *2019 18th European Control Conference (ECC)*. June 2019, pp. 2990–2995.

[73] Tommaso Fedullo, Federico Tramarin, and Stefano Vitturi. "The Impact of Rate Adaptation Algorithms on Wi-Fi-Based Factory Automation Systems". In: *Sensors* 20.18 (2020). ISSN: 1424-8220. DOI: 10.3390/s20185195. URL: https://www.mdpi.com/1424-8220/20/18/5195.

[74] Shancang Li, Li Da Xu, and Shanshan Zhao. "5G Internet of Things: A Survey". en. In: *Journal of Industrial Information Integration* 10 (June 2018), pp. 1–9. ISSN: 2452414X. DOI: 10.1016/j.jii.2018.01.005.

[75] Roberto Maldonado, Anders Karstensen, Guillermo Pocovi, et al. "Comparing Wi-Fi 6 and 5G Downlink Performance for Industrial IoT". In: *IEEE Access* 9 (2021), pp. 86928–86937. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3085896.

[76] Shalitha Wijethilaka and Madhusanka Liyanage. "Survey on Network Slicing for Internet of Things Realization in 5G Networks". en. In: *IEEE Communica-*

tions Surveys & Tutorials 23.2 (2021), pp. 957–994. ISSN: 1553-877X, 2373-745X. DOI: 10.1109/COMST.2021.3067807.

[77] Tommaso Fedullo, Alberto Morato, Federico Tramarin, et al. "Adaptive LoRaWAN Transmission exploiting Reinforcement Learning: the Industrial Case". In: *2021 IEEE International Workshop on Metrology for Industry 4.0 IoT (MetroInd4.0 IoT)*. 2021, pp. 671–676. DOI: 10.1109/MetroInd4.0IoT51437.2021.9488498.

[78] "IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks–Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications". In: *IEEE Std 802.11-2020 (Revision of IEEE Std 802.11-2016)* (2021), pp. 1–4379. DOI: 10.1109/IEEESTD.2021.9363693.

[79] F. Tramarin, A. K. Mok, and S. Han. "Real-Time and Reliable Industrial Control Over Wireless LANs: Algorithms, Protocols, and Future Directions". In: *Proceedings of the IEEE* 107.6 (June 2019), pp. 1027–1052.

[80] A. Kamerman and L. Monteban. "WaveLAN®-II: A high-performance wireless LAN for the unlicensed band". In: *Bell Labs Technical Journal* 2.3 (1997), pp. 118–133.

[81] Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti. "IEEE 802.11 Rate Adaptation: A Practical Approach". In: *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. MSWiM '04. Venice, Italy: Association for Computing Machinery, 2004, pp. 126–134. ISBN: 1581139535.

[82] S. Vitturi, L. Seno, F. Tramarin, et al. "On the Rate Adaptation Techniques of IEEE 802.11 Networks for Industrial Applications". In: *IEEE Transactions on Industrial Informatics* 9.1 (Feb. 2013), pp. 198–208.

[83] F. Tramarin, S. Vitturi, and M. Luvisotto. "A Dynamic Rate Selection Algorithm for IEEE 802.11 Industrial Wireless LAN". In: *IEEE Transactions on Industrial Informatics* 13.2 (Apr. 2017), pp. 846–855.

[84] Qing Li, Qianlin Tang, Iotong Chan, et al. "Smart Manufacturing Standardization: Architectures, Reference Models and Standards Framework". In: *Computers in Industry* 101 (Oct. 2018), pp. 91–106. ISSN: 01663615. DOI: 10.1016/j.compind.2018.06.005.

[85] R. Schlesinger, A. Springer, and T. Sauter. "Concept for the coexistence of standard and Real-time Ethernet". In: *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. June 2018, pp. 1–10. DOI: 10.1109/WFCS.2018.8402344.

[86] D. Dietrich, D. Bruckner, G. Zucker, et al. "Communication and Computation in Buildings: A Short Introduction and Overview". In: *IEEE Transactions on Industrial Electronics* 57.11 (Nov. 2010), pp. 3577–3584. ISSN: 1557-9948. DOI: 10.1109/TIE.2010.2046570.

[87] J. Imtiaz, J. Jasperneite, and S. Schriegel. "A proposal to integrate process data communication to IEEE 802.1 Audio Video Bridging (AVB)". In: *ETFA2011*. Sept. 2011, pp. 1–8. DOI: 10.1109/ETFA.2011.6059004.

[88] F. Zezulka, P. Marcon, Z. Bradac, et al. "Time-Sensitive Networking as the Communication Future of Industry 4.0". In: *IFAC-PapersOnLine* 52.27 (2019). 16th IFAC Conference on Programmable Devices and Embedded Systems PDES 2019, pp. 133–138. ISSN: 2405-8963.

[89] E. Genc and L. F. Del Carpio. "Wi-Fi QoS Enhancements for Downlink Operations in Industrial Automation Using TSN". In: *2019 15th IEEE International Workshop on Factory Communication Systems (WFCS)*. 2019, pp. 1–6.

[90] "IEEE Standard for Local and Metropolitan Area Network–Bridges and Bridged Networks". In: *IEEE Std 802.1Q-2018 (Revision of IEEE Std 802.1Q-2014)* (2018), pp. 1–1993.

[91] *Time-Sensitive Networking (TSN) Task Group official website*. https://1.ieee802.org/tsn/, [Online; Accessed January 4, 2023].

[92] "IEEE Standard for Local and metropolitan area networks - Station and Media Access Control Connectivity Discovery". In: *IEEE Std 802.1AB-2016 (Revision of IEEE Std 802.1AB-2009)* (2016), pp. 1–146. DOI: 10.1109/IEEESTD.2016.7433915.

[93] "IEEE Standard for Local and Metropolitan Area Networks–Timing and Synchronization for Time-Sensitive Applications". In: *IEEE Std 802.1AS-2020 (Revision of IEEE Std 802.1AS-2011)* (2020), pp. 1–421. DOI: 10.1109/IEEESTD.2020.9121845.

[94] "IEEE Standard for Local and Metropolitan Area Networks–Link Aggregation". In: *IEEE Std 802.1AX-2020 (Revision of IEEE Std 802.1AX-2014)* (2020), pp. 1–333. DOI: 10.1109/IEEESTD.2020.9105034.

[95] "IEEE Standard for Local and metropolitan area networks–Frame Replication and Elimination for Reliability". In: *IEEE Std 802.1CB-2017* (2017), pp. 1–102.

[96] "IEEE Standard for Local and Metropolitan Area Networks–Link-local Registration Protocol". In: *IEEE Std 802.1CS-2020* (2021), pp. 1–151. DOI: 10.1109/IEEESTD.2021.9416320.

[97] *P802.1CQ – Multicast and Local Address Assignment*. https://1.ieee802.org/tsn/802-1cq/, [Online; Accessed January 4, 2023].

[98] *P802.1DC – Quality of Service Provision by Network Systems*. https://1.ieee802.org/tsn/802-1dc/, [Online; Accessed January 4, 2023].

[99] "IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture". In: *IEEE Std 802-2014 (Revision to IEEE Std 802-2001)* (2014), pp. 1–74. DOI: 10.1109/IEEESTD.2014.6847097.

[100] *P802.1f – YANG Data Model for EtherTypes*. https://1.ieee802.org/tsn/802f/, [Online; Accessed January 4, 2023].

[101] *P802.1ABcu – LLDP YANG Data Model*. https://1.ieee802.org/tsn/802-1abcu/, [Online; Accessed January 4, 2023].

[102] *P802.1ABdh – Support for Multiframe Protocol Data Units*. https://1.ieee802.org/tsn/802-1abdh/, [Online; Accessed January 4, 2023].

[103] *P802.1ASdm – Hot Standby*. https://1.ieee802.org/tsn/802-1asdm/, [Online; Accessed January 4, 2023].

[104] *P802.1ASdn – YANG Data Model*. https://1.ieee802.org/tsn/802-1asdn/, [Online; Accessed January 4, 2023].

[105] *P802.1CBcv – FRER YANG Data Model and Management Information Base Module*. https://1.ieee802.org/tsn/802-1cbcv/, [Online; Accessed January 4, 2023].

[106] *P802.1CBdb – FRER Extended Stream Identification Functions*. https://1.ieee802.org/tsn/802-1cbdb/, [Online; Accessed January 4, 2023].

[107] "IEEE Standard for Local and metropolitan area networks–Virtual Bridged Local Area Networks Amendment 14: Stream Reservation Protocol (SRP)". In: *IEEE Std 802.1Qat-2010 (Revision of IEEE Std 802.1Q-2005)* (2010), pp. 1–119.

[108] "IEEE Standard for Local and Metropolitan Area Networks - Virtual Bridged Local Area Networks Amendment 12: Forwarding and Queuing Enhancements for Time-Sensitive Streams". In: *IEEE Std 802.1Qav-2009 (Amendment to IEEE Std 802.1Q-2005)* (2010), pp. C1–72.

[109] "IEEE Standard for Local and metropolitan area networks–Media Access Control (MAC) Bridges and Virtual Bridged Local Area Networks–Amendment 18: Enhanced Transmission Selection for Bandwidth Sharing Between Traffic Classes". In: *IEEE Std 802.1Qaz-2011 (Amendment to IEEE Std 802.1Q-2011 as amended by IEEE Std 802.1Qbe-2011, IEEE Std 802.1Qbc-2011, and IEEE Std 802.1Qbb-2011)* (2011), pp. 1–110.

[110] "IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks – Amendment 26: Frame Preemption". In: *IEEE Std 802.1Qbu-2016 (Amendment to IEEE Std 802.1Q-2014)* (2016), pp. 1–52.

[111] "IEEE Standard for Local and metropolitan area networks – Bridges and Bridged Networks - Amendment 25: Enhancements for Scheduled Traffic". In: *IEEE Std 802.1Qbv-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, and IEEE Std 802.1Q-2014/Cor 1-2015)* (2016), pp. 1–57.

[112] "IEEE Standard for Local and metropolitan area networks— Bridges and Bridged Networks - Amendment 24: Path Control and Reservation". In: *IEEE Std 802.1Qca-2015 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qcd-2015 and IEEE Std 802.1Q-2014/Cor 1-2015)* (2016), pp. 1–120.

[113] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks – Amendment 31: Stream Reservation Protocol (SRP) Enhancements and Performance Improvements". In: *IEEE Std 802.1Qcc-2018 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018)* (2018), pp. 1–208.

[114] "IEEE Standard for Local and metropolitan area networks–Bridges and Bridged Networks–Amendment 29: Cyclic Queuing and Forwarding". In: *IEEE 802.1Qch-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd(TM)-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, IEEE Std 802.1Qbz-2016, and IEEE Std 802.1Qci-2017)* (2017), pp. 1–30.

[115] "IEEE Standard for Local and metropolitan area networks–Bridges and Bridged Networks–Amendment 28: Per-Stream Filtering and Policing". In: *IEEE Std 802.1Qci-2017 (Amendment to IEEE Std 802.1Q-2014 as amended by IEEE Std 802.1Qca-2015, IEEE Std 802.1Qcd-2015, IEEE Std 802.1Q-2014/Cor 1-2015, IEEE Std 802.1Qbv-2015, IEEE Std 802.1Qbu-2016, and IEEE Std 802.1Qbz-2016)* (2017), pp. 1–65.

[116] "IEEE Standard for Local and metropolitan area networks–Bridges and Bridged Networks–Amendment 30: YANG Data Model". In: *IEEE Std 802.1Qcp-2018 (Amendment to IEEE Std 802.1Q-2018)* (2018), pp. 1–93.

[117] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks - Amendment 34:Asynchronous Traffic Shaping". In: *IEEE Std 802.1Qcr-2020 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018, IEEE Std 802.1Qcc-2018, IEEE Std 802.1Qcy-2019, and IEEE Std 802.1Qcx-2020)* (2020), pp. 1–151. DOI: 10.1109/IEEESTD.2020.9253013.

[118] "IEEE Standard for Local and Metropolitan Area Networks–Bridges and Bridged Networks Amendment 33: YANG Data Model for Connectivity Fault Management". In: *IEEE Std 802.1Qcx-2020 (Amendment to IEEE Std 802.1Q-2018 as amended by IEEE Std 802.1Qcp-2018, IEEE Std 802.1Qcc-2018, and IEEE Std 802.1Qcy-2019)* (2020), pp. 1–123. DOI: 10.1109/IEEESTD.2020.9212765.

[119] *P802.1Qcj – Automatic Attachment to Provider Backbone Bridging (PBB) services*. https://1.ieee802.org/tsn/802-1qcj/, [Online; Accessed January 4, 2023].

[120] *P802.1Qcw – YANG Data Models for Scheduled Traffic, Frame Preemption, and Per-Stream Filtering and Policing*. https://1.ieee802.org/tsn/802-1qcw/, [Online; Accessed January 4, 2023].

[121] *P802.1Qcz – Congestion Isolation*. https://1.ieee802.org/tsn/802-1qcz/, [Online; Accessed January 4, 2023].

[122] *P802.1Qdd – Resource Allocation Protocol*. https://1.ieee802.org/tsn/802-1qdd/, [Online; Accessed January 4, 2023].

[123] *P802.1Qdj – Configuration Enhancements for Time-Sensitive Networking*. https://1.ieee802.org/tsn/802-1qdj/, [Online; Accessed January 4, 2023].

[124] "ISO/IEC/IEEE International Standard - Amendment 5: Specification and Management Parameters for Interspersing Express Traffic". In: *ISO/IEC/IEEE 8802-3:2017/Amd.5:2017(E)* (2018), pp. 1–62.

[125] Sergei Lupashin, Angela Schöllig, Michael Sherback, et al. "A simple learning strategy for high-speed quadrocopter multi-flips". en. In: *2010 IEEE International Conference on Robotics and Automation*. Anchorage, AK: IEEE, May

2010, pp. 1642–1648. ISBN: 978-1-4244-5038-1. DOI: 10.1109/ROBOT.2010.5509452. (Visited on 12/22/2021).

[126] Mingming Guo, Feng Wang, Fei Peng, et al. "Design of Distributed Network Clock-Synchronization for Swarm UAV". en. In: *2020 International Conference on Computing and Data Science (CDS)*. Stanford, CA, USA: IEEE, Aug. 2020, pp. 194–197. ISBN: 978-1-72817-106-7. DOI: 10.1109/CDS49703.2020.00046. (Visited on 12/23/2021).

[127] A. Neumann, L. Wisniewski, R. S. Ganesan, et al. "Towards integration of Industrial Ethernet with 5G mobile networks". In: *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. 2018, pp. 1–4.

[128] Ruben M. Sandoval, Antonio-Javier Garcia-Sanchez, Joan Garcia-Haro, et al. "Optimal Policy Derivation for Transmission Duty-Cycle Constrained LPWAN". In: *IEEE Internet of Things Journal* 5.4 (2018), pp. 3114–3125. DOI: 10.1109/JIOT.2018.2833289.

[129] *ETSI EN 300 220-1 V3.1.1 (2017-02): Short Range Devices (SRD) operating in the frequency range 25 MHz to 1 000 MHz; Part 1: Technical characteristics and methods of measurement*. https://www.etsi.org/deliver/etsi_en/300200_300299/30022001/03.01.01_60/en_30022001v030101p.pdf, [Online; Accessed January 4, 2023]. 2017.

[130] Emiliano Sisinni, Paolo Ferrari, Dhiego Fernandes Carvalho, et al. "LoRaWAN Range Extender for Industrial IoT". In: *IEEE Transactions on Industrial Informatics* 16.8 (2020), pp. 5607–5616. DOI: 10.1109/TII.2019.2958620.

[131] Frank Loh, Noah Mehling, and Tobias Hoßfeld. "Towards LoRaWAN without Data Loss: Studying the Performance of Different Channel Access Approaches". In: *Sensors* 22.2 (2022). ISSN: 1424-8220. DOI: 10.3390/s22020691.

[132] Carolynn Bernier, François Dehmas, and Nicolas Deparis. "Low Complexity LoRa Frame Synchronization for Ultra-Low Power Software-Defined Radios". In: *IEEE Transactions on Communications* 68.5 (2020), pp. 3140–3152. DOI: 10.1109/TCOMM.2020.2974464.

[133] Kostas Peppas, Kamil Staniec, and Michał Kowal. "LoRa Performance under Variable Interference and Heavy-Multipath Conditions". In: *Wireless Communications and Mobile Computing* (2018). DOI: 10.1155/2018/6931083.

[134] Wei Yin, Peizhao Hu, and Jadwiga Indulska. "Rate control in the mac80211 framework: Overview, evaluation and improvements". In: *Computer Networks* 81 (2015), pp. 289–307.

[135] F. Tramarin, S. Vitturi, M. Luvisotto, et al. "On the Use of IEEE 802.11n for Industrial Communications". In: *IEEE Transactions on Industrial Informatics* 12.5 (Oct. 2016), pp. 1877–1886.

[136] T. Sauter, J. Jasperneite, and L. Lo Bello. "Towards new hybrid networks for industrial automation". In: *2009 IEEE Conference on Emerging Technologies Factory Automation*. 2009, pp. 1–8.

[137] J. Robert, J.–P. Georges, E. Rondeau, et al. "Minimum Cycle Time Analysis of Ethernet–Based Real–Time Protocols". In: *International Journal of Computer Communications & Control* 7.4 (Sept. 2012), pp. 743–757.

[138] R. Zurawski. *Industrial Communication Technology Handbook*. Taylor & Francis Ltd, 2017.

[139] Vinko Erceg. "IEEE P802.11 Wireless LANs TGn Channel Models". In: 2004.

[140] L. Schumacher, K. I. Pedersen, and P. E. Mogensen. "From antenna spacings to theoretical capacities - guidelines for simulating MIMO systems". In: *The 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*. Vol. 2. Sept. 2002, 587–592 vol.2.

[141] *ADMIN-4D project official website.* https://www.t2i.it/events/event/admin-4d-additive-manufacturing-industry-4-0-as-innovation-driver/. [Online; accessed January 04, 2023].

[142] G. Callebaut, G. Ottoy, and L. van der Perre. "Cross-Layer Framework and Optimization for Efficient Use of the Energy Budget of IoT Nodes". In: *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. 2019, pp. 1–6.

[143] Amalinda Gamage, Jansen Christian Liando, Chaojie Gu, et al. "LMAC: Efficient Carrier-Sense Multiple Access for LoRa". In: *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: Association for Computing Machinery, 2020. ISBN: 9781450370851.

[144] Luca Beltramelli, Aamir Mahmood, Patrik Österberg, et al. "LoRa Beyond ALOHA: An Investigation of Alternative Random Access Protocols". In: *IEEE Transactions on Industrial Informatics* 17.5 (2021), pp. 3544–3554. DOI: 10.1109/TII.2020.2977046.

[145] Morgan O'Kennedy, Thomas Niesler, Riaan Wolhuter, et al. "Practical evaluation of carrier sensing for a LoRa wildlife monitoring network". In: *IFIP Networking Conference (Networking)*. 2020, pp. 614–618.

[146] Jeferson Rodrigues Cotrim and João Henrique Kleinschmidt. "LoRaWAN Mesh Networks: A Review and Classification of Multihop Communication". In: *Sensors* 20.15 (2020). ISSN: 1424-8220.

[147] Congduc Pham. "Investigating and experimenting CSMA channel access mechanisms for LoRa IoT networks". In: *IEEE Wireless Communications and Networking Conference (WCNC)*. 2018, pp. 1–6. DOI: 10.1109/WCNC.2018.8376997.

[148] Luca Leonardi, Lucia Lo Bello, Filippo Battaglia, et al. "Comparative Assessment of the LoRaWAN Medium Access Control Protocols for IoT: Does Listen before Talk Perform Better than ALOHA?" In: *Electronics* 9.4 (2020). ISSN: 2079-9292. DOI: 10.3390/electronics9040553.

[149] Yih-Chung Tham, Xiang Li, Tien Y. Wong, et al. "Global Prevalence of Glaucoma and Projections of Glaucoma Burden through 2040: A Systematic Review and Meta-Analysis". In: *Ophthalmology* 121.11 (2014), pp. 2081–2090. ISSN: 0161-6420.

[150] Ahmed Javed, Mohamed Loutfi, Stephen Kaye, et al. "Interobserver reliability when using the Van Herick method to measure anterior chamber depth". eng. In: *Oman Journal of Ophthalmology* 10.1 (Apr. 2017), pp. 9–12. ISSN: 0974-620X. DOI: 10.4103/ojo.OJO_142_2014.

[151] Ivano Riva, Eleonora Micheletti, Francesco Oddone, et al. "Anterior Chamber Angle Assessment Techniques: A Review". In: *Journal of Clinical Medicine* 9.12 (Nov. 2020), p. 3814.

[152] Joan Gispets, Genìs Cardona, Miriam Verdú, et al. "Sources of variability of the van Herick technique for anterior angle estimation". eng. In: *Clinical & Experimental Optometry* 97.2 (Mar. 2014), pp. 147–151. ISSN: 1444-0938. DOI: 10.1111/cxo.12094.

[153] Priya L. Dabasia, David F. Edgar, Ian E. Murdoch, et al. "Noncontact Screening Methods for the Detection of Narrow Anterior Chamber Angles". eng. In: *Investigative Ophthalmology & Visual Science* 56.6 (June 2015), pp. 3929–3935. ISSN: 1552-5783. DOI: 10.1167/iovs.15-16727.

[154] Myra Leung, Sammie Soo Ok Kang, Jason Turuwhenua, et al. "Effects of illumination and observation angle on the van Herick procedure". en. In: *Clinical and Experimental Optometry* 95.1 (Jan. 2012), pp. 72–77. ISSN: 1444-0938. DOI: 10.1111/j.1444-0938.2011.00646.x. (Visited on 10/25/2020).

[155] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1409.1556.

[156] Yingying Wang, Yibin Li, Yong Song, et al. "The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition". In: *Applied Sciences* 10.5 (2020), p. 1897.

[157] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[158] Giovanni Peserico, Tommaso Fedullo, Alberto Morato, et al. "SNR-based Reinforcement Learning Rate Adaptation for Time Critical Wi-Fi Networks: Assessment through a Calibrated Simulator". In: *2021 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*. 2021, pp. 1–6. DOI: 10.1109/I2MTC50364.2021.9460075.

[159] Bruce Pickett, Tarlochan Sidhu, Scott Anderson, et al. "Reducing Outage Durations Through Improved Protection and Autorestoration in Distribution Substations". In: *IEEE Transactions on Power Delivery* 26.3 (2011), pp. 1554–1562. DOI: 10.1109/TPWRD.2011.2114899.

[160] Richard Hunt, Byron Flynn, and Terry Smith. "The Substation of the Future: Moving Toward a Digital Solution". In: *IEEE Power and Energy Magazine* 17.4 (2019), pp. 47–55. DOI: 10.1109/MPE.2019.2908122.

[161] Gert Rietveld, Jean-Pierre Braun, Ricardo Martin, et al. "Measurement Infrastructure to Support the Reliable Operation of Smart Electrical Grids". In: *IEEE Transactions on Instrumentation and Measurement* 64.6 (2015), pp. 1355–1363. DOI: 10.1109/TIM.2015.2406056.

[162] "IEEE/IEC International Standard - Measuring relays and protection equipment - Part 118-1: Synchrophasor for power systems measurements". In: *IEC/IEEE 60255-118-1:2018* (2018), pp. 1–78. DOI: 10.1109/IEEESTD.2018.8577045.

[163] "IEC International Standard - Communication networks and systems for power utility automation – All Parts". In: *IEC 61850:2022 SER* (2022), pp. 1–7892. DOI: 10.1109/IEEESTD.2016.7479438.

[164] Paolo Castello, Paolo Ferrari, Alessandra Flammini, et al. "A New IED With PMU Functionalities for Electrical Substations". In: *IEEE Transactions on Instrumentation and Measurement* 62.12 (2013), pp. 3209–3217. DOI: 10.1109/TIM.2013.2270921.

[165] Davide Della Giustina, Alessio Dedè, Giuliana Invernizzi, et al. "Smart Grid Automation Based on IEC 61850: An Experimental Characterization". In: *IEEE Transactions on Instrumentation and Measurement* 64.8 (2015), pp. 2055–2063. DOI: 10.1109/TIM.2015.2415131.

[166] Marco Agustoni, Paolo Castello, and Guglielmo Frigo. "Phasor Measurement Unit With Digital Inputs: Synchronization and Interoperability Issues". In: *IEEE Transactions on Instrumentation and Measurement* 71 (2022), pp. 1–10. DOI: 10.1109/TIM.2022.3175052.

[167] Mubashir Husain Rehmani, Martin Reisslein, Abderrezak Rachedi, et al. "Integrating Renewable Energy Resources Into the Smart Grid: Recent Developments in Information and Communication Technologies". In: *IEEE Transactions on Industrial Informatics* 14.7 (2018), pp. 2814–2825. DOI: 10.1109/TII.2018.2819169.

[168] Afaf Taik, Boubakr Nour, and Soumaya Cherkaoui. "Empowering Prosumer Communities in Smart Grid with Wireless Communications and Federated Edge Learning". In: *IEEE Wireless Communications* 28.6 (2021), pp. 26–33. DOI: 10.1109/MWC.017.2100187.

[169] Van-Giang Nguyen, Karl-Johan Grinnemo, Javid Taheri, et al. "A Deployable Containerized 5G Core Solution for Time Critical Communication in Smart Grid". In: *2020 23rd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*. 2020, pp. 153–155. DOI: 10.1109/ICIN48450.2020.9059397.

[170] Dick Carrillo, Charalampos Kalalas, Petra Raussi, et al. "Boosting 5G on Smart Grid Communication: A Smart RAN Slicing Approach". In: *IEEE Wireless Communications* (2022), pp. 1–8. DOI: 10.1109/MWC.004.2200079.

[171] "IEC Technical Report - Communication networks and systems for power utility automation - Part 1: Introduction and overview". In: *IEC TR 61850-1:2013* (2013), pp. 1–7892. DOI: 10.1109/IEEESTD.2016.7479438.

[172] Marco Pignati, Lorenzo Zanni, Paolo Romano, et al. "Fault Detection and Faulted Line Identification in Active Distribution Networks Using Synchrophasors-Based Real-Time State Estimation". In: *IEEE Transactions on Power Delivery* 32.1 (2017), pp. 381–392. DOI: 10.1109/TPWRD.2016.2545923.

[173] *OMNeT++ Official WebPage*. https://omnetpp.org/. [Online; Accessed January 4, 2023].

[174] *Simu5g official website*. http://simu5g.org/. [Online; Accessed January 4, 2023].

[175] Giovanni Nardini, Dario Sabella, Giovanni Stea, et al. "Simu5G–An OMNeT++ Library for End-to-End Performance Evaluation of 5G Networks". In: *IEEE Access* 8 (2020), pp. 181176–181191. DOI: 10.1109/ACCESS.2020.3028550.

[176] Giovanni Nardini, Giovanni Stea, and Antonio Virdis. "Scalable Real-Time Emulation of 5G Networks With Simu5G". In: *IEEE Access* 9 (2021), pp. 148504–148520. DOI: 10.1109/ACCESS.2021.3123873.

[177] Giovanni Nardini, Giovanni Stea, Antonio Virdis, et al. "Using Simu5G as a Realtime Network Emulator to Test MEC Apps in an End-To-End 5G Testbed". In: *2020 IEEE 31st Annual International Symposium on Personal, Indoor and Mobile Radio Communications*. 2020, pp. 1–7. DOI: 10.1109/PIMRC48278.2020.9217177.

[178] Martin Klaus Müller, Fjolla Ademaj, Thomas Dittrich, et al. "Flexible multi-node simulation of cellular mobile communications: the Vienna 5G System Level Simulator". English. In: *EURASIP J. Wireless Commun. Netw.* 2018.1 (Sept. 2018). DOI: 10.1186/s13638-018-1238-7.