الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et populaire
وزارة التعليم العالي و البحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Département : Génie Electrique Et Informatique Industrielle

End of study project for the graduation of engineer in
Electronics.

Specialty
Embedded Systems

Design and implementation of an IoT system for indoor air quality monitoring and prediction

Realised by

Mohamed Nadir CHERFIA & Djallel BRAHMIA.

Jury members:

President	Mme.DjamilaBENDOUDA	MCB	ENST
Supervisor	Mme.Leila ABBAD	MAA	ENST
Examinator	Mme.Lamia HABANI	MAA	ENST
Examinator	Mme .Imane DROUICHE	MCB	USTHB

Academic year 2021 -2022

Dedication

“

This work is a fruit of countless and arduous sacrifices. Through the effort, this work is heartily and proudly dedicated to the people who serve as an inspiration. From parents and family, to classmates and circle of friends whom-extended their help in the midst of problems while doing this work

To the faculty and staff of Ecole Nationale Supérieure de Technologie. Above all, to our God Almighty who showered us Hisblessings in our everyday lives, especially for the strength, courage, patience, wisdom,time, and guidance in realization of this work

”

Acknowledgment

This end of study project that we consider as the biggest and most interesting project we have done in our academic lives, searching the topic and understanding every part of it , the conception and the realisation and even the writing of the report was a real challenge which took us several months, we received a lot of support from ALLAH then some people during this process.

First of all, we would like to praise almighty ALLAH for giving us the power to realize this project.

We would like to express our sincere thanks to the members of the jury for the honor they have given us by taking the time to read and evaluate this work.

We would thank our supervisor, **Mme.Leila ABBAD** who supported us, and helped us during the different phases of the project whose expertise was invaluable in formulating the research questions and methodology. Your insightful feedback pushed us to sharpen our thinking and brought our work to a higher level. We would also thank her very much for supporting us and guiding us through every hard counter in our path to finish our project.

We would also give a big thanks to **Mme.K.REBAI** who helped us fixing a lot of issues during the process of realising this project.

We would also address our thanks to **M.Rabah Lali** who shared his expertise with us, guided us and answered our questions during the realisation of this project

Our thanks is also to our families, and friends who supported us mentally and physically and giving us the courage and energy, and laughter when we most need it thank you a lot.

Contents

Dedication . . . .

Acknowledgment . . . .

General introduction **1**

1 Generalities about the Internet of Things **3**
1.1 The IoT,what is it?(Internet of Things)? 4
1.2 IoT architecture 4
1.2.1 Edge devices 5
1.2.2 Gateways and Networks 5
1.2.3 Management Service Layer 6
1.2.4 Application Layer 6
1.3 Internet of Things Communications Models 7
1.3.1 Device-to-Device Communications 7
1.3.2 Device-to-Cloud Communications 7
1.3.3 Device-to-Gateway Model 8
1.3.4 Back-End Data-Sharing Model 9
1.3.5 Internet of Things Communications Models Summary 10
1.4 Choice of an IoT network 11
1.4.1 Personal Network 12
1.4.2 Building Network 12
1.4.3 LPWAN Networks 12
1.5 IoT Protocols 13
1.5.1 Physical layer 14
1.5.1.1 IEEE802.15.4 PHY 14
1.5.2 MAC layer 14
1.5.2.1 IEEE802.15.4E MAC 14
1.5.2.2 ZigBee 15
1.5.2.3 Time Synchronized Mesh Protocol (TSMP) 15
1.5.3 Convergence layer 15
1.5.3.1 6LoWPAN 15
1.5.4 Transport layer 16
1.5.4.1 RPL 16
1.5.5 Session/Application layer 16
1.5.5.1 MQTT 16
1.6 IoT CHALLENGES AND IMPEDIMENTS 16
1.6.1 Addressing plans 17

1.6.2 Big Data . . . 17
1.6.3 Energy Consumption . . . 18
1.6.4 Devices/Links Heterogeneity . . . 18
1.6.5 Transmission Media . . . 19
1.6.6 Security . . . 19
1.6.7 Quality of service (QOS) . . . 19
1.7 IoT Applications . . . 20
1.7.1 Smart Health Care . . . 20
1.7.2 Smart Homes . . . 21
1.7.3 Retail . . . 21
1.7.4 Smart Transportation . . . 21
1.7.5 Industrial Control . . . 22
1.7.6 Smart Agriculture . . . 22
1.7.7 Air Quality Monitoring . . . 22
1.8 IoT and AI blended . . . 23
1.9 Conclusion . . . 23

2 Time series forecasting models . . . 24
2.1 Introduction . . . 25
2.2 An investigation of the autoregressive integrated moving average family for Seasonal Time Series Data . . . 25
2.2.1 Preliminaries . . . 25
2.2.2 Backshift notation . . . 26
2.2.3 Autoregressive moving average models . . . 26
2.2.4 Moving average models . . . 27
2.2.5 Autoregressive moving average (ARMA) models . . . 27
2.2.6 Time series patterns . . . 28
2.2.6.1 Trend . . . 28
2.2.6.2 Seasonality . . . 28
2.2.7 integrated autoregressive moving average models . . . 28
2.2.8 seasonal ARIMA models . . . 28
2.2.9 ACF and PACF plots . . . 28
2.3 a review of deep learning techniques for Time series forecasting . . . 29
2.3.1 Artificial Intelligence, Machine Learning, and Deep Learning . . . 29
2.3.1.1 What Is Machine Learning? . . . 29
2.3.1.2 Whats is Deep leaning? . . . 29
2.3.2 The Bright future of deep learning for time series forecasting . . . 30
2.3.3 Review of recurrent neural networks: Long short-term memory cells and network architectures . . . 31
2.3.3.1 Perceptron and Learning Rules . . . 31
2.3.3.2 Linear Separability . . . 32
2.3.3.3 The Delta Learning Rule . . . 33
2.3.3.4 The Sigmoid Threshold Unit . . . 33
2.3.3.5 The hyperbolic tangent Threshold Unit . . . 34
2.3.3.6 Feed-Forward Neural Networks and Backpropagation . . . 35
2.3.3.7 Recurrent Neural Networks . . . 35

2.3.3.8 LSTM Cells and Their Variants . . . 36
2.3.3.9 An original LSTM: No Forget Gate . . . 36
2.3.3.10 LSTM with a Forget Gate . . . 36
2.3.3.11 LSTM architecture with a peephole connection . . . 37
2.3.3.12 Bidirectional recurrent neural networks . . . 37
2.3.4 An Introduction to Convolutional Neural Networks . . . 38
2.3.5 Basic CNN components . . . 38
2.3.5.1 Convolutional Layer . . . 38
2.3.5.2 Pooling layer . . . 39
2.3.5.3 Fully-connected layer . . . 40
2.3.6 Conv-LSTM Networks for time series prediction . . . 40
2.3.7 TimeDistributed layer . . . 40
2.4 Conclusion . . . 41

3 Conception of the proposed system . . . 42
3.1 Introduction . . . 43
3.2 System requirements analysis . . . 43
3.2.1 Functional requirements . . . 43
3.2.2 Non-Functional requirements . . . 43
3.2.3 Vision of the proposed solution . . . 43
3.2.3.1 Hardware needs . . . 44
3.2.3.2 Software needs . . . 44
3.2.4 How does the system work? . . . 44
3.3 Hardware description . . . 46
3.3.1 Raspberry PI3 . . . 46
3.3.1.1 Defintion . . . 46
3.3.2 Esp32 . . . 46
3.3.2.1 Definition . . . 46
3.3.3 sensors . . . 47
3.3.3.1 PMSA00I . . . 47
3.3.3.2 SEN0035 . . . 47
3.3.3.3 BME680 . . . 48
3.3.3.4 MQ135 . . . 49
3.3.4 Electronic schematics . . . 49
3.4 Software description . . . 50
3.4.1 Visual Studio Code . . . 50
3.4.2 Platform io . . . 50
3.4.3 mosquitto . . . 50
3.4.4 Express JS . . . 50
3.4.5 My-SQL . . . 50
3.4.6 NODE RED . . . 51
3.4.7 Github . . . 52
3.4.7.1 Version control . . . 52
3.4.7.2 Git . . . 52
3.4.8 Docker . . . 52
3.4.9 Python . . . 52

3.4.10 Jupyter Notebook . . . 53
3.5 Conclusion . . . 53

4 Realisation and Results . . . 54
4.1 Introduction . . . 55
4.2 Raspberry Pi3 . . . 55
4.2.1 Installing the operating system . . . 55
4.2.2 Installing Git . . . 55
4.2.3 installing Docker . . . 56
4.2.3.1 Configuring mosquitto . . . 59
4.2.3.2 Configuring mysql . . . 60
4.2.3.3 Configuring NODE RED . . . 61
4.2.4 Configuring node js . . . 62
4.3 Edge device . . . 63
4.3.1 Prediction Model . . . 64
4.3.1.1 DATA PREPROCESSING . . . 64
4.3.1.2 THE HYBRID CNN-LSTM MODEL . . . 65
4.3.1.3 THE SARIMA Model . . . 65
4.4 Results . . . 65
4.4.1 Data reading . . . 66
4.4.2 Monitoring . . . 67
4.4.3 Data storing . . . 67
4.4.4 Prediction . . . 68
4.4.4.1 Neural network . . . 68
4.4.4.2 SARIMA Model . . . 68
4.5 The prototype . . . 69
4.6 Conclusion . . . 70

general Conclusion . . . 71

Appendices . . . 77

A Définitions . . . 78

B Tables . . . 80

C Figures . . . 82

D Features and specs . . . 88

ملخص . . . .

List of Figures

1.1 IoT architecture. 5
1.2 Example of device-to-device communication model. 7
1.3 Example of device-to-cloud communication model. 8
1.4 Example of device-to-gateway communication model. 9
1.5 Back-end data sharing model diagram. 10
1.6 Iot Communication Networks. 11
1.7 Decision tree for IoT communication networks. 13
1.8 IoT protocols. 14
1.9 IoT CHALLENGES AND IMPEDIMENTS. 17
1.10 Big Data Challenges. 18
1.11 Security challenges. 19
1.12 IoT's applications.(Sobin 2020) 20
1.13 Example of a smart health application. 20
1.14 Smart home system.(Sobin 2020) 21

2.1 Two examples of results obtained using autoregressive models with varying parameters (Hyndman et al. 2018) 26
2.2 Two examples of results obtained using moving average models with varying parameters (Hyndman et al. 2018) 27
2.3 The relationship between artificial intelligence, machine learning, and deep learning. (Kelleher 2019) 30
2.4 Topological illustration of a simple neural network (Kelleher 2019) 30
2.5 The general layout of the most basic type of artificial neuron, known as a perceptron. Simple perceptrons are restricted to learn separable linear functions. (Staudemeyer et al. 2019) 32
2.6 Representations of the Boolean functions OR and XOR. The figures demonstrate that the OR function is linearly separable, and that the XOR function is not. (Staudemeyer et al. 2019) 33
2.7 The sigmoid threshold unit is designed to represent non-linear functions. Its output is a continued function of its input, which is comprised between 0. (Staudemeyer et al. 2019) 34
2.8 The Tanh threshold unit is designed to represent non-linear functions. hyperbolic tangent function offers better recognition accuracy than any other functions. 35
2.9 Elman neural network(Staudemeyer et al. 2019). 36
2.10 The Original LSTM architecture(Yu et al. 2019). 36
2.11 The LSTM architecture with a forgetting gate(Yu et al. 2019). 37
2.12 LSTM architecture with a peephole connection(Yu et al. 2019). 37

2.13 Kernel dragging over data and extracting features . . . 38
2.14 Convolution layer slides the filter over a given input. Output is the summation of an element by the element matrix multiplication of the filter and receptive field (Albawi et al. 2017) . . . 39
2.15 Example of (2,2) max pooling . . . 40
2.16 Fully-connected layer . . . 40

3.1 The IoT System's Architecture . . . 45
3.2 ESP32 board . . . 46
3.3 PMSA003I sensor . . . 47
3.4 SEN0335 sensor . . . 48
3.5 Electronic Schematics of the edge device . . . 49
3.6 A NODE RED flow example . . . 51

4.1 Raspberry Pi imager . . . 55
4.2 Containers running in our docker . . . 58
4.3 Docker architecture of our system . . . 59
4.4 NODE RED flows creation . . . 61
4.5 MQTT configuration in NODE RED . . . 61
4.6 Edge device algorithm architecture . . . 63
4.7 year-to-year components . . . 64
4.8 ACF and PACF plots . . . 64
4.9 Edge device with sensors circuit . . . 65
4.10 NODE RED dashboard . . . 67
4.11 A data sample from our stored data in a My-SQL data base . . . 67
4.12 The loss function chart. . . . 68
4.13 Forecast for more than 100 days. . . . 68
4.14 Forecasting result for 8 months. . . . 69
4.15 System's prototype1.0 . . . 69

C.1 TAB 1 NODE RED dashboard monitoring . . . 82
C.2 TAB 2 NODE RED dashboard monitoring . . . 83
C.3 TAB 3 NODE RED dashboard monitoring . . . 83
C.4 TAB 4 NODE RED dashboard monitoring . . . 84
C.5 Visual Studio Code Interface . . . 85
C.6 Platform io Interface in VS Code . . . 85
C.7 Network structure of the proposed hybrid CNN-LSTM model . . . 86
C.8 Internal network structure of the proposed hybrid CNN-LSTM model generated by anaconda platform. . . . 87

List of Tables

3.1 The effects of carbon dioxide and TVOC on the human body 48

4.1 statistical description of PM2.5 concentration values 64

B.1 Major indoor pollutants and emission sources(Farhan et al. 2018b) 80
B.2 Outdoor sources of major indoor air pollutants(Farhan et al. 2018b) 81

Abbreviations

IoT	*Internet of Things*
ALG	*Application Layer Gateway*
LPWAN	*Low Power Wide Area Network*
uIP	*microIP*
LPLN	*Low Power Lossy Network*
IETF	*Internet Engineering Task Force*
MQTT	*Message Queuing Telemetry Transport*
AMQP	*Advanced Message Queuing Protocol*
DDS	*Data-Distribution Service*
LLAP	*Lightweight Local Automation Protocol*
LAN	*Local Area Network*
PAN	*Personal Area Netwok*
UWB	*Ultra Wide Band*
WAN	*Wide Area Network*
WSNs	*Wireless sensors Networks*
QOS	*Quality Of Service*
HITL	*Humans In The Loop*
EPA	*Environmental Protection Agency*

DODAG *Destination Oriented Directed Acyclic Graph*

JS *Java Scripts*

VS Code *Visual Studio Code*

DBMS *Data Base Management System*

RDBMS *Relational Data Base Management System*

API *Applicaion Programming Interface*

LoRa *Long Range*

CSS *Chirp Spread Spectrum*

TSMP *Time-Synchronized Mesh Protocol*

FFD *Full Function Devices*

RFD *Reduced Function Devices*

UDP *User Datagram Protocol*

AI *Artificial Intelligence*

ARIMA *Autoregressive Integrated Moving Average*

SARIMA *Seasonal Autoregressive Integrated Moving Average*

ARMA *Autoregressive Moving Average*

ACF *Autocorrelation Function*

PACF *Partial Autocorrelation Function*

RNN *Recurrent neural network*

ANN *Artificial neural network*

LSTM *Long short term memory*

FFNN *F*eed Forward Neural Networks

CNN convolutional neural network

DNN *D*eep neural network

General introduction

Introduction

The huge development of the Internet of Things (IoT) made it a buzz-subject for everyone interested in new technologies and for the ones working in the research area.Further,with the appearance of developed sensors and devices making interesting networks the IoT has became a true treasure for data analytic.Thus, a significant number of applications has adopted this technology such as in smart health where the doctor has the ability to get information about the patient from the sensor implemented on his body, smart cities where prediction of accidents and traffic flow in cities can be efficiently monitored, industrial production can also be significantly improved by effective prediction of machine operation and also for predicting the quality of air for indoor or outdoor environment which can also be effectively monitored.

Due to the last pandemic of COVID 19 a lot of studies showed that the lockdown applied by the majority of countries produced significant changes in air quality and it has improved, this got the attention of people and they started to rethink about the environment pollution, the quality of air we are breathing and how the particulates (e.g:PM2.5,PM10) carried by air could be a serious danger to our health. More thing that we should point interest to is the indoor air quality and how its really affecting our respiratory system and our health in general. Air quality differ between countries and regions due to several factors,but how can we track the quality of air we are breathing?

That issues leads us to think about how we can get an information about pollutants that aren't visible for human eyes such as the very tiny particulates matter which are a mixture of solid and liquid particles that are suspended in the air and have a diameter less than 2.5 micrometers (PM2.5) giving them the ability to travel deep throw respiratory track, reaching the lungs and the blood stream and can cause a very dangerous problems for those who have breathing and heart issues, the monoxide carbon also can be mortal in indoor environment and a lot of other pollutants which have no color nor smell.

Here where new technologies showed up into scene to propose solution providing devices which can measure pollutants present in the air.

That is where we proposed our solution essentially based on two major concepts: Internet of Things and Artificial Intelligence.

Objectives

The aim of this end of study project is providing a system which gives the ability of monitoring and predicting the air quality in indoor environment. Thus,we focused on designing this system from a hardware to the software implementation to monitor and predict and to provide a data set for a future researches.These proposals are developed to provide a solution for supervising the quality of air which could be used in many fields such as hospitals, pharmaceutical industries ..etc.

The following are the main points of the work carried out in this end of study project

-Propose a system that is able to collect data from the physical to the digital world and monitor it .

-Prediction of the main factors in data which gives us the idea about the quality of air.

-Collecting the data and storing it in a MY-SQL database.

Organization of the dissertation

This dissertation is organized into 04 chapters :

In the first chapter(Generalities about the Internet of Things) we first defined the indoor air quality and then presented generalities about the Internet of Things.

In the second chapter , we have presented several statistical and deep learning techniques that we can use to make a proper prediction for particles of diameter less then 2.5μm

In the third chapter(Conception) we studied the system requirements analysis and described the software and the hardware sides.

In the fourth chapter (Realisation and results) we presented the process making the system and results obtained.

Chapter 1

Generalities about the Internet of Things

Introduction

To date, most Internet applications focus on providing information, interaction, and entertainment for humans. However, with the widespread deployment of networked, intelligent sensor technologies, an Internet of Things (IoT) is steadily evolving, much like the Internet decades ago. In the future, hundreds of billions of smart sensors and devices will interact with one another without human intervention, on a Machine-to-Machine (M2M) basis. They will generate an enormous amount of data at an unprecedented scale and resolution, providing humans with information and control of events and objects even in remote physical environments.

In this chapter ,and also we will see generalities about IoT and its architecture, Communications models, how to chose an IoT network, IoT protocols, IoT challenges and impediments, Application we pointing out the importance of taking care of indoor air quality and some of the main pollutants and how important is to embed Artificial Intelligence (AI) with IoT.

1.1 The IoT,what is it?(Internet of Things)?

The term “Internet of Things” is more or less new, the concept of combining computers and networks to monitor and control devices has been around for decades. The term “Internet of Things” (IoT) was first used in 1999 by British technology pioneer Kevin Ashton to describe a system in which objects in the physical world could be connected to the Internet and that could in wireless or in a wired technology. The Internet connectivity and advanced data analysis combined with consumer’s durable goods , cars and trucks , sensors and other everyday objects changed the way we live , work and play.Projections for the impact of IoT on the Internet and economy are magnificent , with some expectations as many as 100 billion connected IoT devices and a global economic impact of more than $11 trillion by 2025. (Rose et al. 2015)

1.2 IoT architecture

The IOT architecture is made up of different layers of technologies. It is used to illustrate how the different technologies are linked to each other and to communicate scalability, modularity and configuration of IoT deployments in different scenarios. Figure 1.1 shows the detailed architecture of the IOT. The functionality of each layer is described below

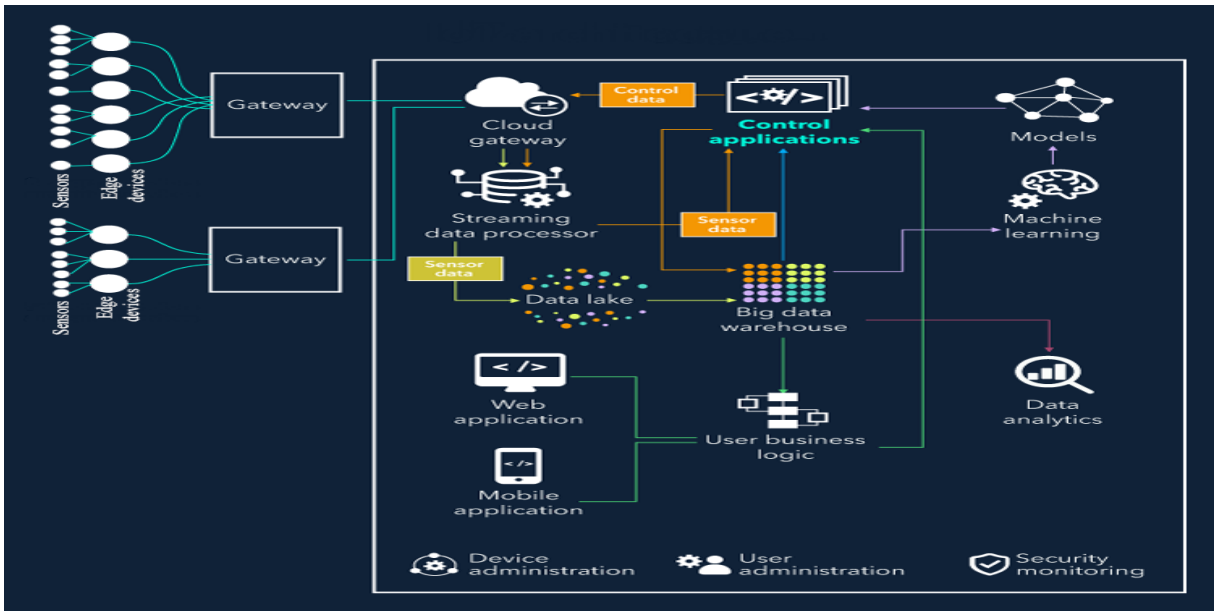


Figure 1.1: IoT architecture.

1.2.1 Edge devices

The base layer is made up of edge devices composed usually of sensors and actuators which enable the interconnection between the physical and digital world giving the abilities of collecting and processing a real-time information.There are many types of sensors for different purposes having the capacity to take measurements such as temperature, air quality, speed, humidity, pressure, etc.In some cases sensors may have a degree of memory allowing them to record a certain number of measurements.A sensor measurement is a conversion of a physical property into a signal that can be readable by an instrument.Sensors are grouped according to their unique purpose like environmental sensors, body sensors and vehicle telematics sensors, etc.

The majority of edge devices connected to sensors require connectivity to gateways which can be in a form of a Local Area Network (LAN) such as Ethernet and Wi-Fi connections or Personal Area Network (PAN) such as ZigBee, Bluetooth and Ultra Wide Band (UWB).For sensors that doesn't require connectivity to aggregators, their connectivity to back-end server/application can be granted using Wide Area Network (WAN) such as GSM, GPRS and LTE.Low power and low data rate connectivity rate sensors typically form network commonly known as wireless sensors networks (WSNs).WSNs are making popularity as they can receive far more sensor nodes while retaining adequate battery life and covering large areas.(Patel et al. 2016)

1.2.2 Gateways and Networks

A huge amount of data will be produced by smart sensors that has the ability to communicate directly to a network, but in most used cases that huge amount of data is produced by edge devices which serve as entry or exit points and control the flow of data at the boundary or perimeter between two networks which will help bridging the

digital and the physical worlds. and this needs a robust and high performance wired or wireless network infrastructure as a transport medium.Now-days networks which are usually tied up with different types of protocols have been used to support machine-to-machine (M2M) networks and their applications.With the need to serve a wider range of IoT services and applications such as high speed transnational services, context-aware applications, etc.Networks with various technologies and access protocols are needed to work with each other in a heterogeneous configuration.These networks can be in the form of a private, public or a hybrid models and are built to support the communication requirements for latency, bandwidth or security.(Patel et al. 2016)

1.2.3 Management Service Layer

The management service makes the processing of information through analysis, security controls process modeling and device management.

The business and process rule engine are one of the most important features of the management service layer.IoT brings connection and interaction of objects and systems together presenting the information in a form of events or contextual data such as temperature, humidity,air quality and particulates in the air , etc.Some of these events requires filtering or routing to post-processing systems such as capturing of periodic sensory data,while others requires response to the immediate situation such as reacting to emergencies on patient's health conditions. The rule engines support the formulation of decision logics and trigger interactive and automated processes to enable a more responsive IOT system.

In the analytic area,multiple analytics tools are used to extract relevant information from big data sets and to be processed at a much faster rate.Analytics such as inmemory analytics allows large volumes of data to be cached in random access memory (RAM) rather than stored in physical disks. In-memory analysis reduces data query time and increases decision-making speed. Continuous analysis is another form of analytics where the analysis of data, considered as moving data, must be data in motion, is done in real time for decisions to be made in a timely manner.

Data management is the ability to manage data information flows.Using data management in the management service layer , information can be accessed , integrated and controlled.

Security must be applied across the entire dimension of the IOT architecture, from the smart object layer to the application layer. System security prevents hacking and compromise of the system by unauthorized personnel thus reducing the possibility of risks.

1.2.4 Application Layer

The IoT application covers "smart" environment/spaces in domains such as: Transportation, Building, City Life-Style, Retail , Agriculture, Factory, Supply chain, Emergency, Healthcare, user interaction, culture and tourism, Environment and Energy.

1.3 Internet of Things Communications Models

IoT devices are everywhere and will enable circulating intelligence in the future. For operational perception, it is important and useful to understand how different IoT devices communicate with each other. The communication patterns used in IoT have great value. IoTs allow people and objects to be connected at any time, in any space, with anything and anyone, using any network and any service.(Kulkarni et al. 2017)

1.3.1 Device-to-Device Communications

The device-to-device communication model represents two or more devices that directly connect and communicate between one another, rather than through an intermediary application server.These devices communicate over multiple types of networks, including IP networks or the Internet.However, these devices often use protocols such as Bluetooth[1], Z-Wave[2] or ZigBee[3] to achieve direct communications between devices.

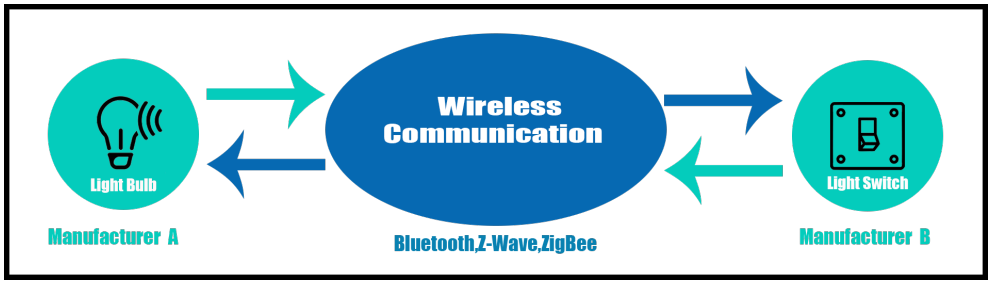


Figure 1.2: Example of device-to-device communication model.

These device-to-device networks permit devices that adhere to a particular communication protocol to communicate and exchange messages to accomplish their function. This communication model is largely used in applications such as home automation systems, which normally use small data packets to communicate between devices with relatively low data rate requirements. Residential IoT devices such as light bulbs, switches, thermostats and door locks, normally send small amounts of information to each other (e.g., a door lock status message or a command to turn on the light) in a home automation scenario.

1.3.2 Device-to-Cloud Communications

In a device-to-cloud communication model, the IoT device directly connects to an Internet cloud service, like an an application service provider to exchange data and to manage message flow. This is an approach that often relies on communication mechanisms, such as traditional wired Ethernet or Wi-Fi connections, to establish a connection between

[1] http://www.bluetooth.com
[2] http://www.z-wave.com
[3] http://www.zigbee.org

the device and the IP network, which eventually connects to the cloud service.(Kulkarni et al. 2017)

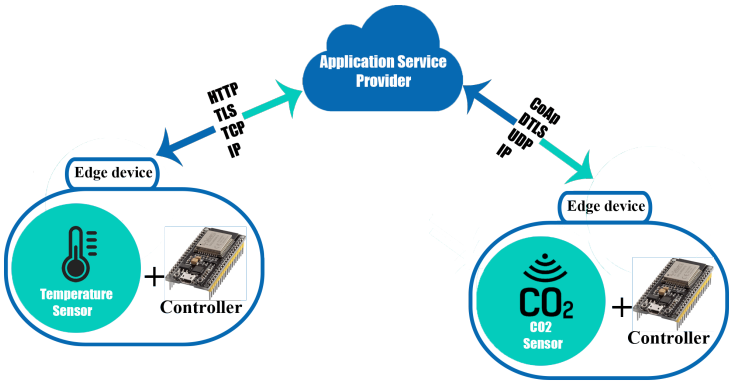


Figure 1.3: Example of device-to-cloud communication model.

This communication model is being used by some consumer IoT devices like the Nest Labs Learning Thermostat[4]. In the case of the Nest Learning Thermostat, the device sends data to a cloud database which it can use to analyze the home's energy use. In addition, this cloud connection gives the user remote access to their thermostat via a smartphone or web interface, and it also supports software updates to the thermostat. Similarly, with Samsung SmartTV technology, the TV uses an Internet connection to transmit the user's viewing information to Samsung for analysis and to enable the TV's interactive voice recognition features. In these cases, the device-to-cloud model adds value to the end user by extending the capabilities of the device beyond its native features.(Kulkarni et al. 2017)

1.3.3 Device-to-Gateway Model

In the gateway device model, or more commonly in the application layer gateway (ALG) device model, the IoT device connects through an ALG service as a conduit to reach a cloud service. IoT device connects through an ALG service as a conduit to reach a cloud service. In simple terms, this means that an application software runs on a local gateway device, which serves as an intermediary between the device and the cloud service and assures security and other functionality such as data or protocol translation. (Kulkarni et al. 2017)

[4] "Meet the Nest Thermostat | Nest." Nest Labs. Web. 31 Aug. 2015. https://nest.com/thermostat/meet-nest-thermostat/



Figure 1.4: Example of device-to-gateway communication model.

There are many forms of this model in consumer devices. In most cases, the local gateway device is a smartphone running an app to communicate with a device and then relay data to a cloud service.This is often the model employed with popular consumer items like personal fitness trackers. These devices do not have the native ability to connect directly to a cloud service, so they usually rely on smartphone app software to serve as an intermediary gateway to connect the fitness device to the cloud.

The other form of this gateway device model is the arrival of ”hub” devices in home automation applications. . These are devices that serve as a local gateway between individual IoT devices and a cloud service, but they can also bridge the interoperability gap between devices themselves. For example, the SmartThings hub is a stand-alone gateway device that has Z-Wave and Zigbee transceivers installed to communicate with both families of devices [5]. It then connects to the SmartThings cloud service, allowing the user to gain access to the devices using a smartphone app and an Internet connection.

This communication model is often used to integrate new smart devices into an existing system with devices that are not naturally interoperable with them. The disadvantage of this approach is that the necessary development of the application layer software and gateway system adds complexity and cost to the overall system.(Kulkarni et al. 2017) The evolution of systems using the device-to-gateway communication model and its larger role in addressing interoperability challenges among IoT devices is still unfolding. (Kulkarni et al. 2017)

1.3.4 Back-End Data-Sharing Model

The back-end data sharing model refers to a communication architecture that allows users to export and analyze smart object data from a cloud service in combination with data from other sources. This approach is an extension of the single device-to-cloud communication model, which can lead to data silos where IoT devices upload data only

[5] “How It Works.” SmartThings, 2015.
http://www.smartthings.com/how-it-works

to a single application service provider. A back-end sharing architecture allows the data collected from single IoT device data streams to be aggregated and analyzed.(Kulkarni et al. 2017)

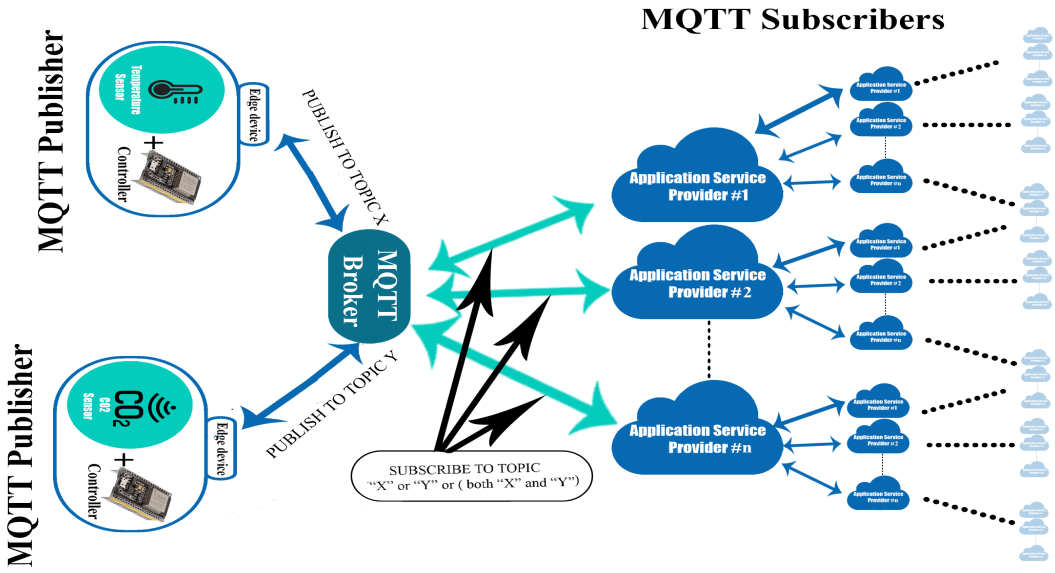


Figure 1.5: Back-end data sharing model diagram.

The back-end data-sharing model suggests a federated cloud services approach[6]. or cloud applications programming interfaces (APIs) are needed to achieve interoperability of smart device data hosted in the cloud.

1.3.5 Internet of Things Communications Models Summary

The four basic communication patterns demonstrate the basic underlying design strategies used to enable IoT devices to have communication. Apart from some technical aspects, the use of these models is largely influenced by the open or proprietary nature of the networked IoT devices.And in the case of the device-to-gateway, its main feature is its ability to overcome the restrictions of proprietary devices in the connection of IoT devices. This means that device interoperability and open standards are key considerations in the design and development of networked IoT systems.

From a general user perspective, these communication models help illustrate the ability of networked devices to add value to the end user. By allowing the user easier access to an IoT device and its data, the overall value of the device is amplified. For example, in three of the four communication models, the devices ultimately connect to data analytic services in a cloud computing setting.By building data communication conduits to the cloud, users and service providers can more easily perform data aggregation, massive

[6] A federated cloud services approach is one that combines the resources of separate cloud service providers to meet a larger business need

data analysis, data visualization and predictive analytics to obtain more value from IoT data in comparison to traditional data silo applications.(Kulkarni et al. 2017)

While the end user is benefiting from efficient communication models, it is worth mentioning that IoT communication models also strengthen technical innovation and open up opportunities for commercial growth. New products and services can be developed to take advantage of IoT data streams that did not exist before, acting as a catalyst for further innovation.

1.4 Choice of an IoT network

The chose of an IoT network is based on differents factors.First of all we can find three types of IoT network:

1)-Personal networks.

2)-Building network.

3)-Open network.

Technically, in addition to the transmission distance, the difference between these networks is also in the frequency bands used and the energy required to use them.

There are many application areas that will be impacted by the emergence of the Internet of Things. The applications can be classified according to network type, availability, coverage, scale, heterogeneity, repeatability user involvement and impact. In the upcoming sections we will see the differents IoT network types.

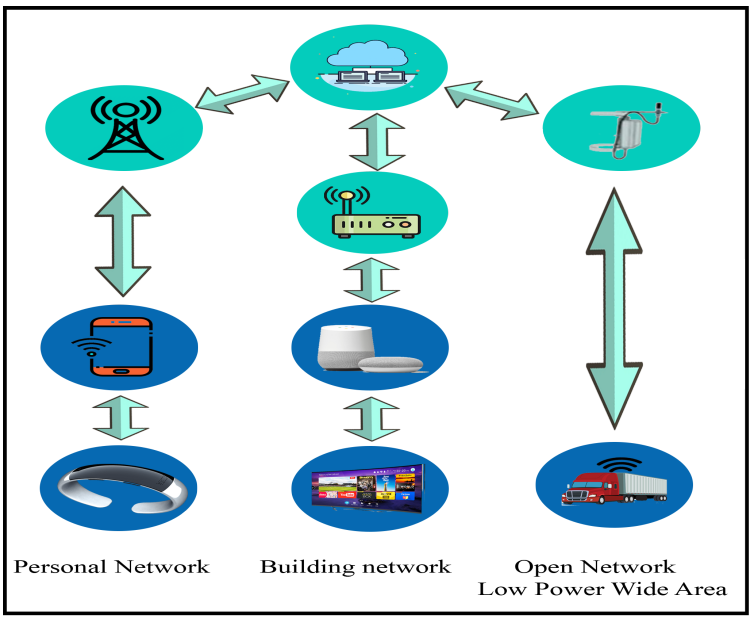


Figure 1.6: Iot Communication Networks.

1.4.1 Personal Network

They are the general public networks attached to the person, oriented around sport wellness, leisure, media, social networks and personal effectiveness. These networks are therefore very often linked to smartphones or to miniaturized and portable devices. The main networks are :

-**Bluetooth (BLE, BLE Smart, etc...)** which allows easy connectivity within a few meters using a communication protocol widely used and available natively on many devices.(Connectwave n.d.)

-**ANT+** which is a proprietary protocol, with many industrial partners and very strongly linked to sports and well-being.(Connectwave n.d.)

1.4.2 Building Network

The main building networks are :

Industrial BLE, derived from Bluetooth for the general public and made reliable for industry, allows the creation of dense networks in warehouses and offices to track assets and sensors, with a sensors, with a typical communication distance of 20 to 200m. Being able to associate tags with beacons, repeaters and routers, these networks allow a wide range of networks and a wide range of applications, including indoor geolocation. of buildings.(Connectwave n.d.)

-**Zigbee**: building mesh network, allowing to connect objects from close to close, based on a proprietary protocol managed by the Zigbee Alliance, often used in home automation to replace electrical wiring (lights, comfort sensors warning sensors).(Connectwave n.d.)

-**Z-Wave**: mesh network quite similar to Zigbee, also based on an alliance of industrialists ZWave Alliance.(Connectwave n.d.)

-**WiFi Hallow** : another mesh network for home automation, also based on IPv6 and a WiFi protocol, a proprietary protocol driven by the WiFi Alliance.(Connectwave n.d.)

1.4.3 LPWAN Networks

These are the new networks (cellular or not) dedicated to connected objects. They are ideal for transmitting simple information from sensors and location to object management platforms, on the cloud or on private networks.

These networks are based on protocols focusing on :

-The very low consumption (autonomy of 5 to 10 years).

-The very long communication distance (several km).

- Good coverage inside buildings.

-The very high density of connectable objects.

-Low operating costs.

The counterpart of these advantages is the limited communication rate and the limi-

tation of the number of bidirectional communications (from the object or to the object). The main LPWAN networks are :

-**LoRa**:LoRa (short for long range) is a spread spectrum modulation technique derived from chirp spread spectrum (CSS) technology.LoRa devices and networks such as the LoRaWAN® enable smart IoT applications that solve some of the biggest challenges facing our planet: energy management, natural resource reduction, pollution control, infrastructure efficiency, and disaster prevention..[7].

-**3GPP : LTE-M, NB-IOT, 5G**.

-**Proprietary or market networks**: ingenu, RPMA, Wimax, Qowisio, Weightless.

Faced with this range of possibilities, it is essential to choose the right communication network. This one will necessarily depend on the connected object.Here is an explanatory diagram to help finding the way around.

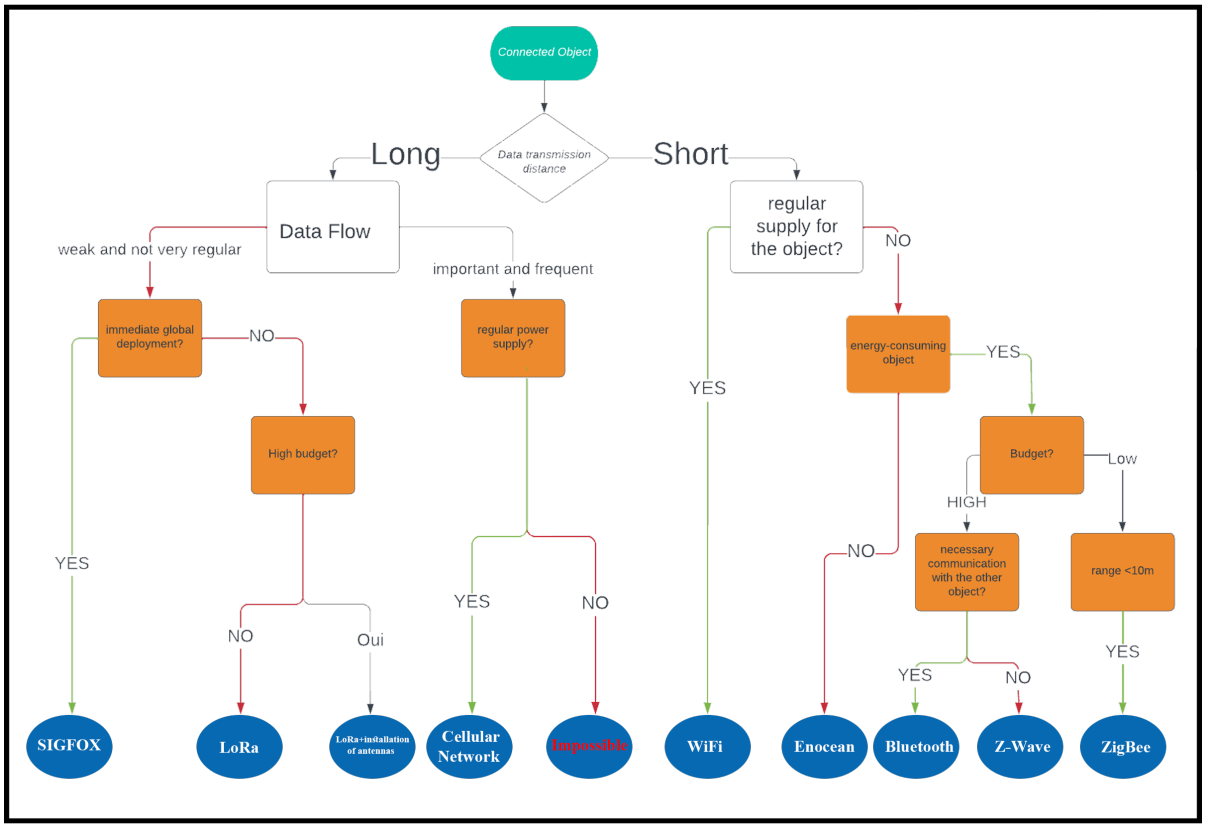


Figure 1.7: Decision tree for IoT communication networks.

1.5 IoT Protocols

IoT protocols must meet the requirements below, consume little power, be highly reliable, and the devices must be able to connect to the Internet.

[7] https://www.semtech.com/lora/what-is-lora,June 2022.

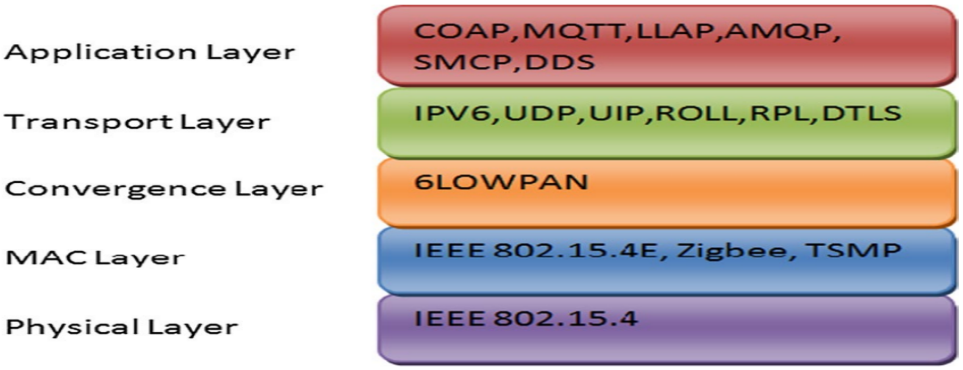


Figure 1.8: IoT protocols.

In the following is the protocol stack and the set of some of the protocols used in each layer.

1.5.1 Physical layer

1.5.1.1 IEEE802.15.4 PHY

This protocol is defined by IEEE 802.12.4 for the physical layer. It supports low power devices at the physical layer or sensor layer. T The main purpose of the protocol is to define the physical layer and the MAC layer for the operation of low-power wireless personal area networks (LR-WPANs). It supports many upper layer protocols, such as ZigBee, Time-Synchronized Mesh Protocol (TSMP), etc. The transfer rate of the base frame is 250 KB/s and supports a communication range of 10 to 100 meters. Other features of the standard include: CSMA/CD to avoid collisions, guaranteed time slots to support real-time communications and security aspects, etc [8].

The types of devices generally are:

(1) Full Function Devices (FFD), which supports any topology and have full functionalities, like routers, coordinators, etc.

(2) Reduced Function Devices (RFD), which can only use star topology and is having with limited functionality, so cannot function as coordinator and sleeps when there is no activity(Ma et al. 2008).

1.5.2 MAC layer

1.5.2.1 IEEE802.15.4E MAC

This protocol is defined for MAC layer of IoT, which can transmit the frames through IEEE802.15.4 PHY channel.There are two modes of operation supported by IEEE802.15.4E MAC layer:

(1) Beacon-mode, in which synchronization is achieved with the help of super frames

[8] https://en.wikipedia.org/wiki/IEEE_802.15.4, Mai 2022.

and CSMA-CA mechanism and achieves dedicated bandwidth and low power consumption.

(2): Nonbeacon-mode, which is the traditional multiple access system and uses CSMA-CA for collision avoidance.(Sobin 2020)

1.5.2.2 ZigBee

is a wireless technology developed to build personal area networks (PAN), with low-power, low-cost, low-bandwidth devices that can be used to develop IoT applications. ZigBee can be used to create mesh networks, in which several paths exist between network nodes. Such a topology permits nodes to dynamically group together and provide ad-hoc routing functionality to network nodes. (Sobin 2020)

1.5.2.3 Time Synchronized Mesh Protocol (TSMP)

A communication protocol used by motes, which are wireless devices with a self-organizing capability.The SMP is synchronized and the communication between the stations uses time slots in a manner similar to TDMA. Unlike ZigBee, TSMP is designed to operate in a noisy environment and uses channel hopping to eliminate interference. And it's also mainly used for reliable and durable applications.(Sobin 2020)

1.5.3 Convergence layer

1.5.3.1 6LoWPAN

Is an IPv6 variant of low power wireless PAN, which allows send/ receive IPv6 packets over IEEE 802.15.4 based networks [9]. For backward compatibility the 6LoWPAN protocol is offered to introduce IPv6 into the IoT. 6LoWPAN allows IPv6 to be used in IoT, so the current network can be easily deployed in IoT. . 6LoWPAN is mainly used for application scenarios, such as home automation, smart meters, etc.As pointed out in (Ma et al. 2008), there are so many advantages of using 6LoWPAN:

(1) Since, IPv6 is the next generation Internet, integrating IPv6 with LR-WPAN will help to design next generation IoT applications.

(2) The vast address space in IPv6 compared to IPv4, meets the requirements of today's IoT and Big data applications.

(3) The feature of stateless and auto-confguration will help the nodes to read the MAC addresses automatically.

[9] https://en.wikipedia.org/wiki/6LoWPAN, Mai 2022.

1.5.4 Transport layer

1.5.4.1 RPL

Is the routing protocol developed for low power and lossy networks. RPL supports multipoint-to-point trafc in IoT networks(Ko et al. 2011). For multipoint-to-point traffic RPL creates a DODAG (Destination Oriented Directed Acyclic Graph) used for routing. . RPLs also have optional security functions to support message authentication using the Advanced Encryption Standard (AES) and RSA signatures for data integrity.(Sobin 2020)

1.5.5 Session/Application layer

1.5.5.1 MQTT

MQTT was originally designed as a message transport for real-time SCADA systems. The MQTT message transport specification does not specify the subject namespace to be used, nor does it define the payload representation of published and/or subscribed data which is intentional within the original specification to provide maximum flexibility across any solution sector that might choose to use MQTT infrastructures which contains in general(Sparkplug cited June 2022):

-**Mqtt Server(s)**
MQTT enabled infrastructure requires that one or more MQTT Servers are present in the infrastructure(Sparkplug cited June 2022).

-**Mqtt Edge of Network (EoN)**
An MQTT Edge of Network (EoN) Node is any compliant MQTT Client application that manages an MQTT Session and provides the physical and/or logical gateway functions required to participate in the Topic Namespace and Payload definitions(Sparkplug cited June 2022).

-**Device/Sensor** The device/sensor represents any physical or logical device connected to the MQTT EoN providing data(Sparkplug cited June 2022).

1.6 IoT CHALLENGES AND IMPEDIMENTS

Although there is a proliferation of IoT applications today, it is still in its infancy and as a result, there are several challenges and limitations which are summarized in FIG 1.9 .

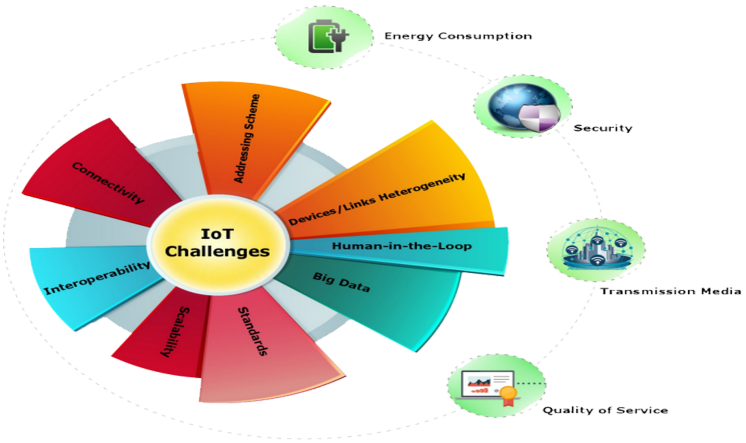


Figure 1.9: IoT CHALLENGES AND IMPEDIMENTS.

In what follows, we will focus on some of the design issues and challenges that is currently hindering IoT to achieve its fullest potential.

1.6.1 Addressing plans

The unique identification of objects is a key issue for the operation and success of IoT applications.IoT applications require thousands of devices to be uniquely classified and remotely managed and controled remotely via the Internet.The few most critical features of creating unique address are reliability, uniqueness, scalability and persistence (Gubbi et al. 2013).These smart devices need an appropriate and unique address that will allow them to communicate with each other and be part of the internet. Internet protocol version 4 (IPv4) uses 32-bit addresses and provides capacity for only 4.3 billion IP addresses which are almost out of addresses. The next generation is IPv6 and it uses 128-bit addresses and has massive address abundance $3.4 * 1038$ or (340 trillion trillion trillion)(Palattella et al. 2013).A group of researchers has designed a lightweight addressing system to solve the IoT heterogeneity problem. In this model, virtual domain and multiple coding were used to implement node addressing. The proposed scheme shows an appropriate and fixable interconnection between WSNs and the IPv6-based Internet over 6LowPAN[10]. This work also presents an addressing scheme combining the cluster-tree algorithm with the AODV routing protocol. It implements an addressing scheme for nodes in the local and wide area of IoT networks. To this end, a unified addressing scheme for IoT is a very popular research area and a great challenge.(Farhan et al. 2018a)

1.6.2 Big Data

IoT harnesses massive amounts of data collected and aggregated via smart objects and it's one of the most prominent features of the IoT.It will be essential to develop techniques that convert these data into usable knowledge. Every two years, data is doubling in size and is expected to reach 44 Zettabytes in the next four years(Farhan et al. 2017).

[10] https://www.researchgate.net/publication/283744525_Research_on_the_Model_of_a_Lightweight_Resource_Addressing,Mai 2022.

The ”5V ” (Value, Velocity, Volume, Variety, and Veracity) are defined in A.1 are important challenges in IoT applications as presented in Fig 1.10.



Figure 1.10: Big Data Challenges.

1.6.3 Energy Consumption

The IoT creates billions of devices and networks of a wide variety of devices connected to the internet. Energy is considered a critical resource for smart IoT devices because most applications are battery powered or use power management techniques.This means it is not prudent to waste the energy by transmission of unneeded data and protocol overheads that existing protocols such as HTTP, TCP, etc.The study in (Farhan et al. 2018c) introduced a new scheduling algorithm for nodes located between two or more sensing fields. SMAC protocol intended the border nodes that consume a large amount of energy and adopted different sleep and listen schedules. These nodes often switch to listening mode due to diverse scheduling. Work has focused on minimizing the energy consumption of the border nodes, thereby extending the lifetime of WSNs.If IoT is to have a sustainable impact, energy consumption must be high on the agenda. A typical scenario where battery runs out of the nodes demanding change in the battery is not feasible since this means battery change in millions of nodes for a particular application. Therefore, the IoT should operate more or less in a disposal mechanism where nodes last as long as possible and are discarded when their life span is over. (Farhan et al. 2018a)

1.6.4 Devices/Links Heterogeneity

Another important feature of the IoT vision is the variety of devices and links, as they will operate on different sets of protocol suite, data formats, etc.In WSNs, most of the sensors are homogeneous, i.e. they have the same power, communication and equal computational capacity.The IoT implements a wide variety of networks, links, and connectivity devices to provide different services. Thus, the heterogeneous nature of links and objects plays a critical role in the interconnection of IoT devices and therefore presents a unique challenge.

1.6.5 Transmission Media

Is the physical path that establishes the connection and carries data from the sender to the receiver.IoT networks use different types of technology to transmit or receive the data such as Bluetooth, Zigbee, LoraWAN, etc. The traditional problems associated with transmission media (e.g., high error rate, bandwidth, fading, inference, etc.) exists for IoT as well. Each transmission medium requires specialized power, network hardware, and bandwidth that must be compatible with that medium. Therefore, optimizing the Transmission Media is a challenge in IoT applications to support and extend the life of the networks.

1.6.6 Security

The problem of security is one of the main challenges of networks over the years. Thus, security, privacy and trust are critical factors for IoT applications as well. When packets are routed through different links and devices to reach the final receiver on the Internet, measures must be taken to ensure that the confidentiality and integrity of the data is maintained.In addition, most IoT devices are low-power devices. established cryptographic solutions cannot be directly applied to the IoT scene.Also, currently, the integration of an application into the network infrastructure focuses solely on the realization of functionality. rather than holistically considering the security requirements when designing the application. This is leaving door open for attacks and hacking attempts. Cybersecurity experts have warned that IoT is one of the most vulnerable technology and they expect more targeted attacks on existing and emerging infrastructures, e.g. data theft, physical injury, DDoS attack, ransomware for smart homes or smart cars, etc.(Farhan et al. 2018a) Four key IoT security challenges can be seen in Fig 1.11:

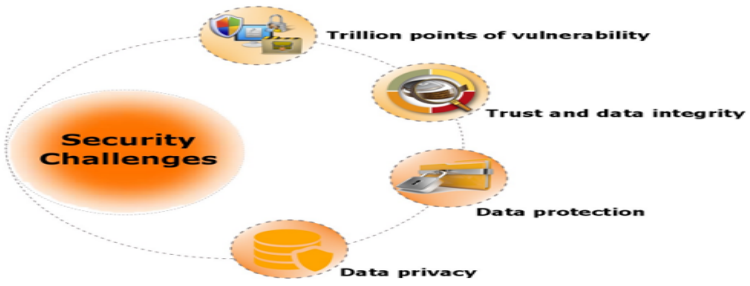


Figure 1.11: Security challenges.

1.6.7 Quality of service (QOS)

In many applications, the data collected must be delivered within a certain timeframe to the intended destination, otherwise the data will be of lesser value.The QoS requirements are met with differentiated services and delay management, packet loss, and bandwidth parameters on a network.These demands become the secret of a successful end-to-end service. Therefore, the quality of the service must be researched and stabilized in order to be implemented.(Farhan et al. 2018a)

1.7 IoT Applications

There are many applications of IoT in various fields such as home, healthcare, industries, retail, transportation, surveillance, etc.

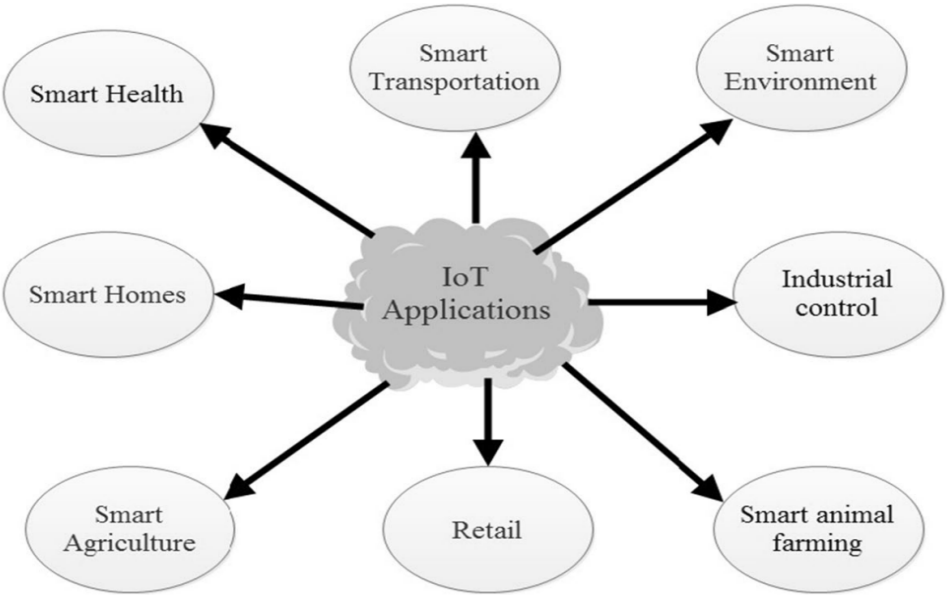


Figure 1.12: IoT's applications.(Sobin 2020)

Some of the IoT applications are listed below.

1.7.1 Smart Health Care

The integration of IoT capabilities into medical devices will improve the quality of services to patients, particularly the elderly who require constant monitoring. IoT can be used to monitor the elderly for fall detection, to monitor temperature conditions inside the refrigerator which stores medications, vaccines, and to monitor the condition of patients in hospitals and at home. Today, the focus on smart health solutions is to develop wearable devices that will help capture medical data.(Sobin 2020)

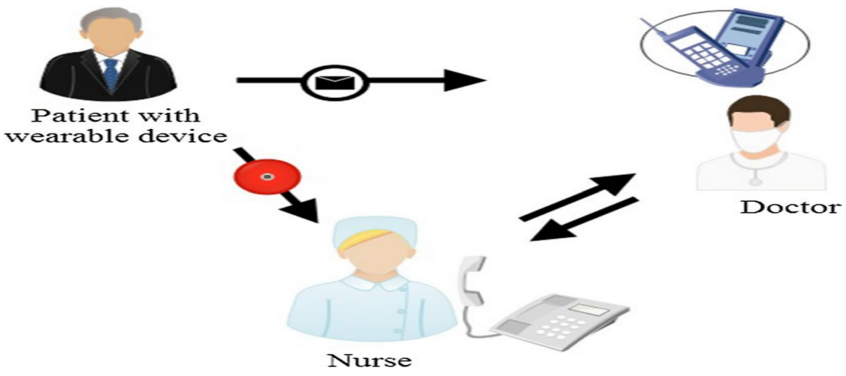


Figure 1.13: Example of a smart health application.

The figure above describes a scenario where a wearable device is attached to the patient, which will send notifcations to nurses or doctors directly, when there is an abnormal behaviour of the patient health. Although the expansion of IoT in healthcare sector is a promising area, there are some issues also to be considered related to data privacy and security.(Sobin 2020)

1.7.2 Smart Homes

In smart homes, we will have the ability to control any device in the home using a laptop or smartphone and the devices will communicate with each other to providing us with useful services.By using smart meters, we can measure energy consumption, and temperature. As shown in Figure 1.14, a smart home is composed of several smart elements, such as light control, temperature control, remote control, camera monitoring, high-speed Internet, smart devices, smart media, and so on.(Sobin 2020)

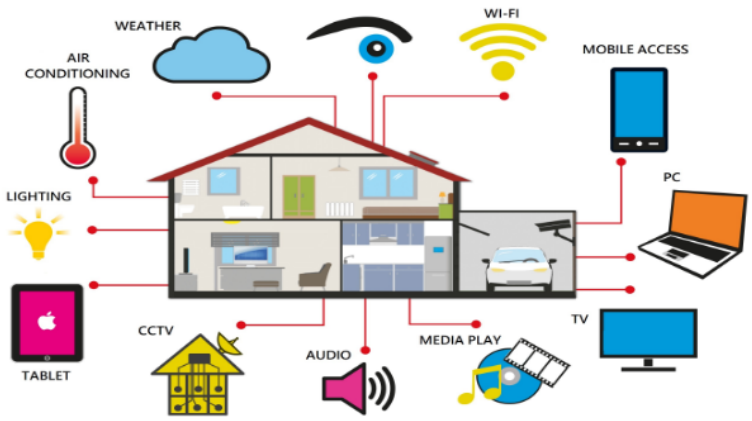


Figure 1.14: Smart home system.(Sobin 2020)

1.7.3 Retail

IoT can be used for product tracking, for monitoring the storage conditions of products. It can be used to give suggestions to customers when shopping based on their habits, but also in the automatic restocking process by assisting in the control and rotation of products in the shelves and warehouses of stores.(Sobin 2020)

1.7.4 Smart Transportation

IoT is used to find the location of items in large warehouses or ports, check the quality conditions during the shipment of goods, monitor the routes taken for delicate goods such as gold, medical drugs, etc. (Sobin 2020)

1.7.5 Industrial Control

In industries, IoT is used to monitor air quality, temperature, ozone presence, and send an alarm in case of emergency. An example is the baggage handling machine used in airports, where sensors detect if the bag is moving on the conveyor belt and stops if none is moving. Luggage is identified on the basis of on the basis of bar-codes.(Sobin 2020)

1.7.6 Smart Agriculture

IoT can be used to monitor soil moisture, control weather conditions to maximize fruit production, controlling humidity and temperature levels to avoid fungi. IoT can monitor weather conditions to predict rain, ice formation, drought, etc. A real-life example of such a smart agricultural system is a smart cattle farm in which information about the movements, behavior, fertility and lactation of cattle is available owners using smart phones, which will help increasing productivity.

1.7.7 Air Quality Monitoring

Until recently, the health effects of indoor air pollution have received relatively little attention from the scientific community. Prior to the 1970s, indoor air quality problems in residences and non-industrial workplaces were occasionally studied, but the level of interest was low.Even today, the bulk of public concern continues to be directed at the health impacts of outdoor pollution.

Indoor air pollutants emanate from a range of sources (Table B.1). They are emitted by the fabric of buildings, and may also be a by-product of the activities that are undertaken within them. Sources can be broadly classified as being associated with the activities of building occupants and other biological sources, the combustion of substances for heating or fuel, and emissions from building materials. For some contaminants, infiltration from outside, either through water, air, or soil, can also be a significant source.(Farhan et al. 2018b) The concentration of a pollutant indoors depends on the relationship between the volume of air contained in the indoor space, the rate of production or release of the pollutant, the rate of removal of the pollutant from the air via reaction or settling, the rate of air exchange with the outside atmosphere, and the outdoor pollutant concentration (Maroni et al. 1995). This can create uncertainties, especially for pollutants with health effects thought to be related to short-term extreme exposures. In the case of recording pollutants from sources such as micro-organisms an opposite problem can apply, whereby monitoring programmes must be operational for a very long time period to provide accurate estimates of exposure (Blomquist 1994). And as in example in the past few years, levels of smog have increased throughout China resulting in the deterioration of air quality, raising worldwide concerns. PM2.5 (particles less than 2.5 micrometers in diameter) can penetrate deeply into the lung, irritate and corrode the alveolar wall, and consequently impair lung function.[11]. It is important to note that outdoor sources may be the main contributors to indoor concentrations of a number of non-biological pollutants

[11] https://www.ncbi.nlm.nih.gov/pmc/,April 2022.

commonly found in indoor air.Some of those pollutants are cited in the table B.2 . One of the most interesting applications now-days is monitoring the quality of the air using sensors to collect data indoor or outdoor like CO2,PM2.5,CO,etc.Those informations could be transformed to a useful data to be monitored like in hospitals and also could be treated with AI using prediction.In our work we will treat this application in details going throw the whole IoT architecture.

1.8 IoT and AI blended

At its core, IoT is about sensors embedded in machines, which offer data streams through Internet connectivity. All IoT-related services inevitably follow five basic steps: create, communicate, aggregate, analyze and act. Undeniably, the value of the "act" depends on the penultimate analysis. Therefore, the precise value of the IoT is determined at the analysis stage. This is where AI technology plays a crucial role [12]. IoT artificial intelligence leads to a broad range of benefits for companies and consumers like a proactive intervention, personalized experience, and intelligent automation.

1.9 Conclusion

In this first chapter,we presented how the indoor air quality could affect our health and which are the principal pollutants from outdoor and indoor environment,also we have presented the general concept of the Internet of Things, the architecture and its uses.
Furthermore,we presented the protocols used in the different layers of the IoT architecture and the bases we should lay on while choosing an IoT type of network,IoT's applications and how the technologies of now days is interested into embedding it with AI and challenges that faces it. All of those concepts were necessary to consume in order to understand better the problematic and better choosing the technologies and avoiding problems in the process of making this project.
In the next chapter we will discuss the Artificial Intelligence part where several models were presented and explained in order to make the data coming from our IoT system usable for prediction.

[12] https://www.clariontech.com/blog/ai-and-iot-blended-what-it-is-and-why-it-mattersp,Mai 2022.

Chapter 2

Time series forecasting models

2.1 Introduction

The classical statistical models have largely been applied to solve air quality prediction challenges. Those approaches are based on the concept of learning from historical data. Some of the notable statistical methods that have been used for air quality forecasting are the autoregressive moving average ARMA , the autoregressive integrated moving average ARIMA and seasonal autoregressive integrated moving average SARIMA . But, with the growing volume and complexity of the data obtained, such methods can no longer satisfy the real demand due to the time taken to learn. As artificial intelligence and big data develop, the use of predictive methods based on machine learning technologies is becoming more popular Artificial neural networks (ANNs) that model the complex non-linear relationships between air pollutant levels and atmospheric parameters. Different ANN structures have been developed to perform air pollution prediction in various areas of study

2.2 An investigation of the autoregressive integrated moving average family for Seasonal Time Series Data

In statistics, Box-Jenkins models are the main time series models. Considering a time series $X(t)$, these models are a useful tool to understand and predict, eventually, the future values of this series

2.2.1 Preliminaries

the time series with a variable of interest is denoted by $x(t)$, the data sets are available for t = 1, 2,..., T , i.e. T is the sample size. The key aspect of a time series, as opposed to cross-sectional data, is that the data $x(t)$ are viewed sequentially, i.e. observation $x(t-1)$ precedes observation $x(t)$, which then precedes $x(t+1)$. In particular, at time $t-1$, the current observation at t is not yet known, while every observation up to and including $t-1$ is recognized. We refer to the history of a time series up to time t - 1 as the set $x(t-1) \Longrightarrow \{x_1, x_2, ..., x_(t-1)\}$. The fact that the time series data are observed in a sequence suggests that there may be information in the set $x_(t-1)$ that can be used to explain or predict $x(t)$. Another, more formal way of expressing the above is as follows. At time $t-1$, x_t is unknown and can be treated as a random variable. The assumption is that the observations may contain useful information about the value of x_t (Gourieroux et al. 1997), for example if the temperature was 7 degrees Celsius in the morning, 10 degrees at lunchtime so it is highly possible to have by evening 12 than to have 38

2.2.2 Backshift notation

The back shift generator B is a helpful tool when it comes to dealing with lags in time series:

$$BX_t = X_{t-1} \tag{2.1}$$

In another sense, B , operating on X_t has the result of delaying the data by one time period. Two applications of B to X_t shift the data by two periods backwards:

$$B(BX_t) = B^2X_t = X_{t-2} \tag{2.2}$$

2.2.3 Autoregressive moving average models

In a multi-regression model, we are predicting the variable of interest using a linear combination of predictors. In an autoregressive model, we predict the variable of interest with a linear combination of previous values of the variable. The word autoregression means a regression of the variable on itself. So, a p-order autoregressive model can be expressed as the following:

$$x_t = c + \phi_1 x_{t-1} + \phi_2 x_{t-2} + ...\phi_p x_{t-p} + \epsilon_t \tag{2.3}$$

where /epsilon is white noise. It is like a multilevel regression but with lagged values of X_t as forecasters. We call this an AR(p), an autoregressive model of p order. Autoregressive

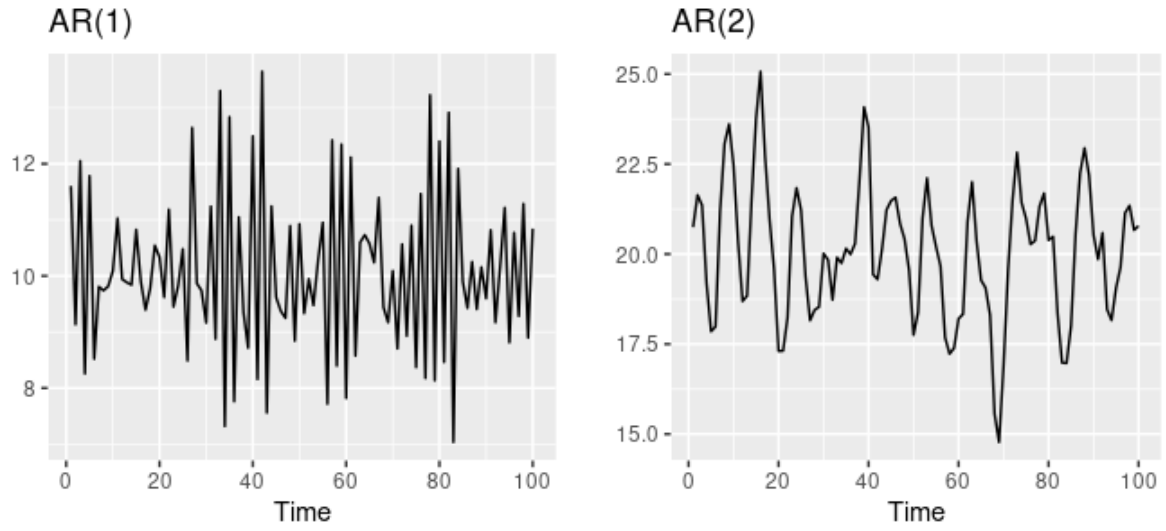


Figure 2.1: Two examples of results obtained using autoregressive models with varying parameters (Hyndman et al. 2018)

models are notably adaptable to deal with a large variety of different time series models. The two series in Fig 2.1 illustrate series from an AR(1) model and an AR(2) model. Varying the ϕ_1....ϕ_p parameters gives different time series models. The variance of the error term ϵ will only alter the size of the series, not the patterns.(Hyndman et al. 2018)

2.2.4 Moving average models

Instead of relying on previous values of the forecast variable in a regression, a moving average model reuses previous prediction errors in a regression-type model.

$$x_t = \epsilon_1\varphi_1 + \epsilon_2\varphi_2 + ... + \epsilon_p\varphi_p \tag{2.4}$$

where ϵ_t is a white noise. We call this an MA(q) model, a moving average model of order q. Of course, we do not observe the readings of ϵ_t because they are random and unpredictable signals that have a significant effect on the time series caused by factors beyond our scope , so it really cannot be a regression in the conventional way. Fig 2.2 illustrates some results of an MA(1) model and an MA(2) model. By varying the φ parameters, we have different time series models(Hyndman et al. 2018)

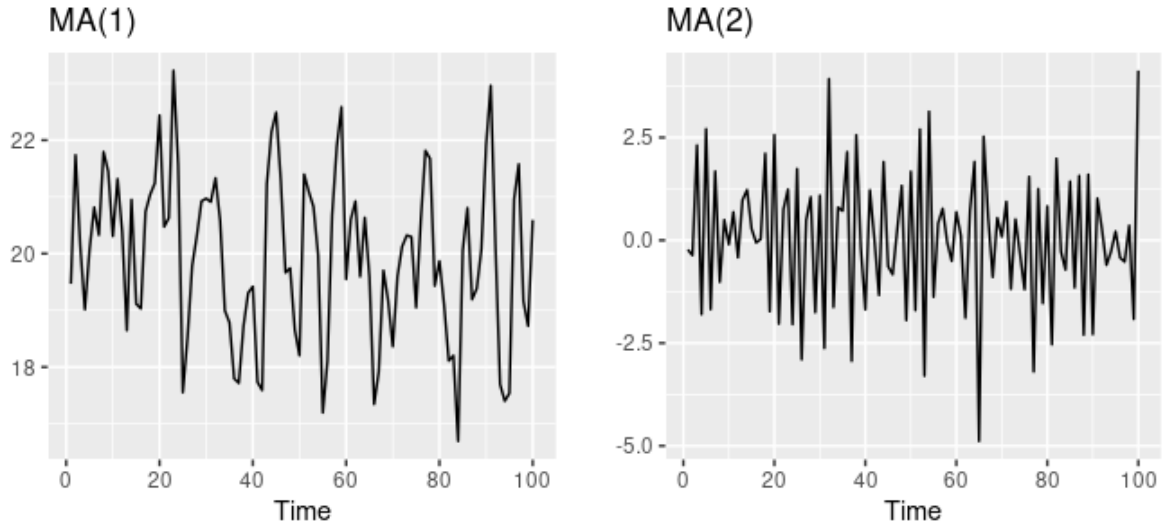


Figure 2.2: Two examples of results obtained using moving average models with varying parameters (Hyndman et al. 2018)

2.2.5 Autoregressive moving average (ARMA) models

Moving average models consider the fact that returns do not necessarily depend only on current information, but also on signals that arrived in a preceding time period.This can happen if new information is only gradually absorbed or reach forecast variable at various points in time by . Therefore, every new event has not just an instant effect, but also a lagged effect. Autoregressive models assumed that there exists a linear relation between the present returns and its own past history (Brandimarte 2013).

These two notions can be used together, the resulting model is called an autoregressive moving average model, ARMA(p,q) :

$$x_t = C + \sum_{l=1}^{p}[\varphi x_{t-l}] + \sum_{l=1}^{p}[\phi\epsilon_{t-l}] \tag{2.5}$$

2.2.6 Time series patterns

2.2.6.1 Trend

A trend is present whenever there is a long-term rise or fall in the pattern of data. It need not necessarily follow a linear path. sometimes a trend is described as "changing direction," when it changes from an uptrend to a downtrend (Hyndman et al. 2018).

2.2.6.2 Seasonality

Seasonal variation results if a time series is influenced by seasonal variables such as the season of the year or the day of the week. Seasonality is ALWAYS of a definite and recognized frequency (Hyndman et al. 2018).

2.2.7 integrated autoregressive moving average models

A time series X_t is referred to as one following an integrated autoregressive moving average model if the dith difference $W_t = \nabla^d X_t$ ($\nabla^d X_t = X_t - X_{t-d}$) is a stationary ARMA process. If Wt is following an ARMA(p,q) model, we would say that Yt is an ARIMA(p,d,q) based process(Cryer et al. 2008). Let us then take an ARIMA(p,1,q) process. Given:
$W_t = X_t - X_{t-1}$, we have

$$W_t = \sum_{l=1}^{p} [\varphi_l W_{t-l}] + \sum_{l=1}^{p} [\phi_l \epsilon_{t-l}] \tag{2.6}$$

or, in terms of the observed series:

$$X_t - X_{t-1} = \phi_1(X_{t-1} - X_{t-2}) + \phi_2(X_{t-2} - X_{t-3}) ... + \phi_p(X_{t-p} - X_{t-p-1}) + \epsilon_t + \varphi_{1t-1} + \varphi_2\epsilon_{t-2} + ... + \varphi_q\epsilon_{t-q} \tag{2.7}$$

that we can generalise it for an ARIMA(p,d,q) as the following :

$$(1 - B^d)X_t = (\phi_1 B + \phi_2 B^2 + ... + \phi_p B^p)(1 - B^d)X_t + \epsilon_t(1 + \varphi_1 B + \varphi_2 B^2 + ...\varphi_q B^q) \tag{2.8}$$

2.2.8 seasonal ARIMA models

We have restricted attention to non-seasonal data and non-seasonal ARIMA models. But ARIMA models are able to handle a large variety of seasonal data as well(Hyndman et al. 2018). A seasonal ARIMA model is made by including extra seasonal terms in the ARIMA models we have viewed so far. It is expressed as the following:

$$(1 - \phi_1 B)(1 - \vartheta_1 B^{12})(1 - B)(1 - B^4)X_t = (1 + \varphi_1 B)(1 + \theta B^{12})\epsilon_t \tag{2.9}$$

2.2.9 ACF and PACF plots

An ACF measure and plot the average Correlation of data points in a time series with earlier measured values in the series for various lag length. A PACF is analogous to an

ACF, with the exception that partial autocorrelation is how much correlation a variable has with a lag of itself that is is not explained by the correlations at all low-order lags. The PACF here is expected to provide you with an idea of the AR lags and ACF of the MA lags(Box et al. 2015).

2.3 a review of deep learning techniques for Time series forecasting

The whole field of artificial intelligence was born during a summer workshop at Dartmouth College in 1956. Research on a number of topics was presented at this workshop, including computer programs capable of learning from examples and neural networks.

2.3.1 Artificial Intelligence, Machine Learning, and Deep Learning

2.3.1.1 What Is Machine Learning?

A machine learning-based algorithm is a search process that aims to determine the optimal function, from a range of possible functions, for understanding the dependencies between features in a data set (Kelleher 2019). Machine learning approaches are traditionally divided into three main categories, depending on the nature of the "signal" or "feedback" available to the learning system:

Supervised learning:
consists of learning through pre-labeled inputs, which are used as goals. For each example of training, there will be a set of input data sets (vectors) with one or more assigned output values that are associated. The goal of this form of learning is to minimize the overall model error by properly computing the output value (O'Shea et al. 2015).

Unsupervised learning:
is different because the training set does not have labels. However, the success is usually measured by the network's ability to minimize or increase an associated cost function(O'Shea et al. 2015)

Semi-supervised learning:
It is a class of machine learning techniques that uses a set of labeled and unlabeled data. It is thus between supervised learning which only uses labeled data and unsupervised learning which only uses unlabeled data. It aims to reveal the underlying distribution of the examples in their description space. It is implemented when data (or labels) are missing (Amihai et al. 2018)

2.3.1.2 Whats is Deep leaning?

Deep learning is a branch of Machine learning that concentrates on building huge neural network models with the ability of making data-based, accurate decisions. Deep learning is specifically suited to contexts of complex and large data sets. Most of today's

online companies and high-end consumer technologies use deep learning
The term deep learning refers to a family of neural network models that have multiple layers of single information Processing programs, also known as neurons, inside the network.(Kelleher 2019).

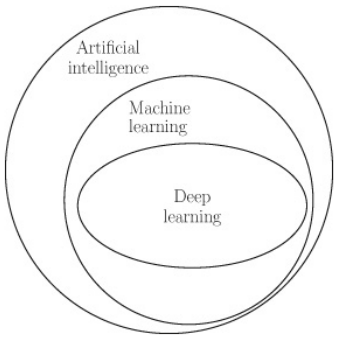


Figure 2.3: The relationship between artificial intelligence, machine learning, and deep learning. (Kelleher 2019)

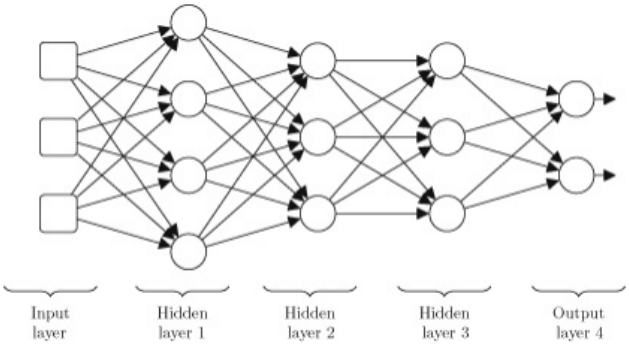


Figure 2.4: Topological illustration of a simple neural network (Kelleher 2019)

2.3.2 The Bright future of deep learning for time series forecasting

Deep learning neural networks are able to learn complex arbitrary input-output mappings automatically and support multi-input and multi-output. These are very promising and powerful features for predicting time series, particularly for problems with complex and non-linear dependencies, multiple-valued inputs, and multi-step forecasts. Such features, as well as the more modern neural network capabilities, can be highly promising, such as the automatic feature learning afforded by convolutional neural networks and the built-in support for sequential data in recurrent neural networks
Forecasting time series is challenging. In contrast to the more simple classification and regression problems, time series problems introduce additional complexity of temporal ordering or dependency between observations. This can be hard because a specialized treatment of the data is necessary during model fitting and evaluation. This temporal

structure can also assist in modeling, by providing additional structure like trends and seasonality that can be exploited to increase model skill(Brownlee 2018). More basic neural networks, such as the multilayer perceptron or MLP, approximate a mapping function of input to output variables. This general skill is of great value for time series for a number of reasons.

Noise resistance:
Neural networks are noise robust to both the input data and the mapping function and can even provide training and prediction in the face of missing values(Dorffner 1996).

Nonlinear:
an additional important feature of neural networks - their powerful ability to perform approximations of random nonlinear functions. This property is of high value, particularly in the field of forecasting (Dorffner 1996)

Multi-step Forecasts:
An arbitrary amount of output values can be specified, which gives direct multi-step and even multi-variable forecasting support(Dorffner 1996).

2.3.3 Review of recurrent neural networks: Long short-term memory cells and network architectures

Recurrent neural networks (RNNs) have been widely implemented in research topics of sequential data, such as text, audio and video. But RNNs that consist of sigma or tanh cells are unable to learn meaningful information from the input data when the input gap is large. By incorporating gate functions into the cell structure, long-term memory (LSTM) may be able to handle the long-term dependency problem very well. Since its introduction, . LSTM has become the main point of focus in deep learning(Yu et al. 2019).

2.3.3.1 Perceptron and Learning Rules

Artificial neural networks are made up of a group of neuron-based threshold switching units. each unit receives a subset of real-valued data inputs and generates a single real-valued output. Based on the connectivity between the neuron threshold units and the parameters of the elements, these networks can provide a complex global behavior.
The simplest type of artificial neurons is known as a perceptron. A perceptron consists of a number of input links, a threshold and a single output link. In some cases, perceptrons also have an internal input, called a bias.A perceptron takes a vector of input values, all of which are weighted by a multiplier, the perceptron learns these weights based on a learning, It sums all the weighted input values and is activated if the resulting value is greater than a predefined value. The output of the perceptron is always Boolean, and it is considered activated if the output is equal to ”1”. The disabled value of the perceptron is ’-1’, the threshold value is, in most cases, ’0’. The process of learning the perceptron is an imitative learning process, referred to as Supervised Learning. During the training stage, the perceptron generates an output and matches it with a value of output obtained from the training data. In the face of misclassification, the perceptron changes the weights accordingly.(Staudemeyer et al. 2019) show that within a certain

amount of time, the perceptron is able to converge to the correct behavior, as long as the learning samples are linearly separable. Convergence is not guaranteed if the training data are not linearly separable. A variety of learning algorithms exists for perceptrons, the most popular of which are the perceptron learning rule and the delta learning rule. Both starts with random weights and ensures convergence to an acceptable hypothesis. acceptable hypothesis.

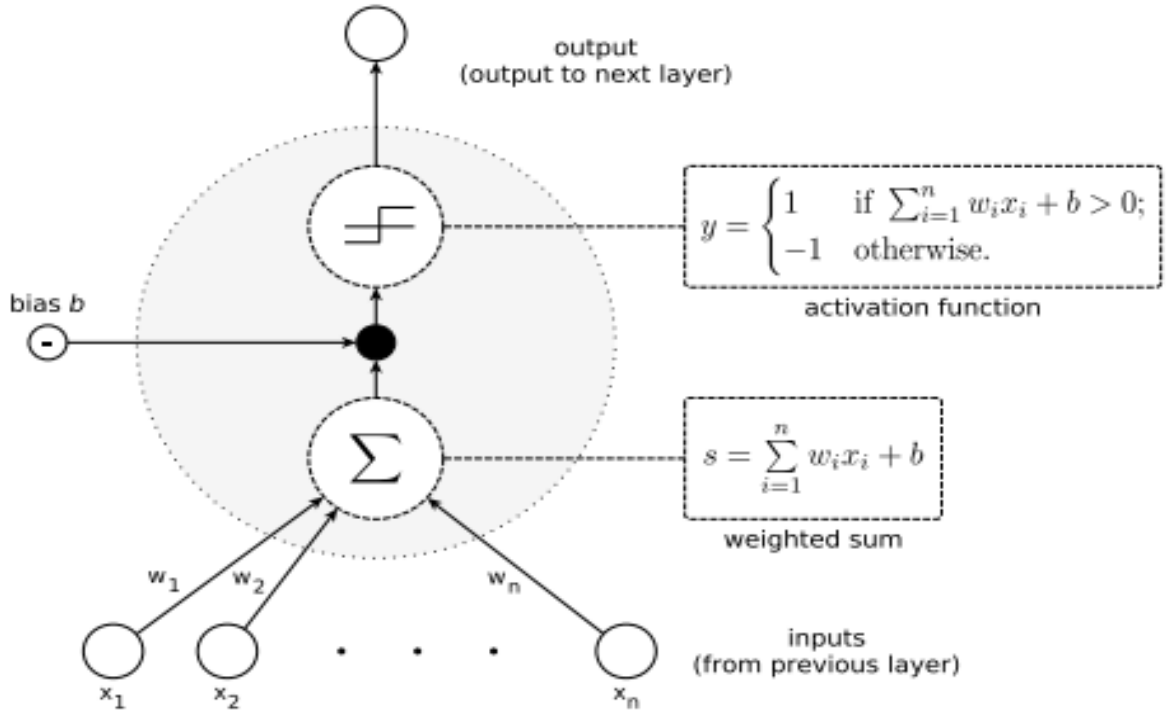
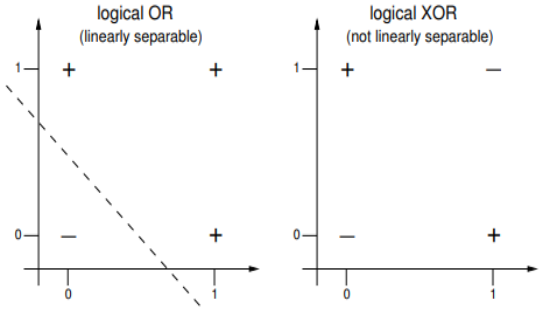


Figure 2.5: The general layout of the most basic type of artificial neuron, known as a perceptron. Simple perceptrons are restricted to learn separable linear functions. (Staudemeyer et al. 2019)

2.3.3.2 Linear Separability

To explain linear separability, it is useful to view the possible inputs to a perceptron on the axes of a 2-D graph. the following figure illustrates the Boolean functions OR and XOR. The OR function is linearly separable, but the XOR function is not. In the figure, pluses are shown for an input where the perceptron has been activated and minuses are shown where it has failed to be triggered. If plus and minus can be entirely divided by a single line, the problem is linearly separable into two dimensions.



Input 1	Input 2	Output
1	1	1
1	0	1
0	1	1
0	0	0

Input 1	Input 2	Output
1	1	0
1	0	1
0	1	1
0	0	0

Figure 2.6: Representations of the Boolean functions OR and XOR. The figures demonstrate that the OR function is linearly separable, and that the XOR function is not. (Staudemeyer et al. 2019)

2.3.3.3 The Delta Learning Rule

The process of training in the hidden layers is called supervised learning. In the Training stage,The output layer generates an output and matches it with a derived output value supplied by the training data. If there is any misclassification, it then changes the weights of the perceptrons (which are the core of the hidden layers) accordingly,within a finite time, the training would converge to the correct behavior, as long as the training data are linearly separable. If the training data are not linearly separable, convergence is not assured.
While there are a range of learning mechanisms, the most commonly used are the learning rule and the delta learning rule. Both begin with a random weight and provide a probability of convergence to an acceptable hypothesis.
The delta learning rule was especially designed to treat linearly separable and linearly non-separable training samples. It also performs error calculations between the computed output and the output data of the training instances, and varies the weights accordingly. The modification of the weights is achieved by using the gradient optimization descent algorithm, which modifies them in the way that produces the fastest downhill descent along the error surface to the lowest possible global error. The delta learning rule is the base for the error back-propagation algorithm(Staudemeyer et al. 2019).

2.3.3.4 The Sigmoid Threshold Unit

The sigmoid threshold unit is a different kind of artificial neuron, quite similar to the perceptron, but using a sigmoid function to calculate the output. The output y is computed by the formula:

$$y = \frac{1}{(1 - e^{l*s})} \tag{2.10}$$

with :

$$s = \sum W_i X_i + b_i \tag{2.11}$$

with b being the bias and l being a fixed positive quantity that defines the steepness of the sigmoid curve. The primary impact on the perceptron is that the output of the sigmoid unit now has more than two different values; now the output is " squashed " by a continuous function between 0 and 1. sigmoid is known as the squashing function, since it mapping a very wide range of inputs to a limited range of outputs.
The benefit of neural networks using sigmoid units is that they are able to represent non-linear functions. Cascaded linear units, such as the perceptron, are limited to the representation of linear functions.

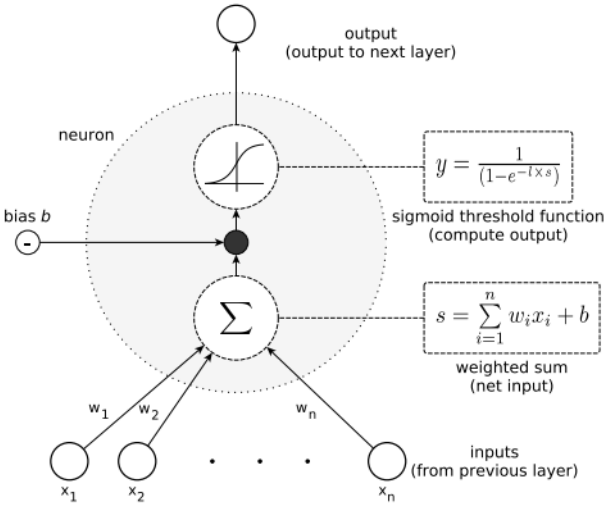


Figure 2.7: The sigmoid threshold unit is designed to represent non-linear functions. Its output is a continued function of its input, which is comprised between 0. (Staudemeyer et al. 2019)

2.3.3.5 The hyperbolic tangent Threshold Unit

This function is conveniently defined as the ratio between hyperbolic sine and cosine functions or stretched to be the ratio of the half-difference and the half-sum of two exponential functions at points x and -x as follows:

$$tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2.12}$$

The hyperbolic tangent function is very similar to the sigmoid function. Its scale outputs between -1 and 1, The Tanh (hyperbolic tangent) function offers better recognition accuracy than any other functions(Karlik et al. 2011).

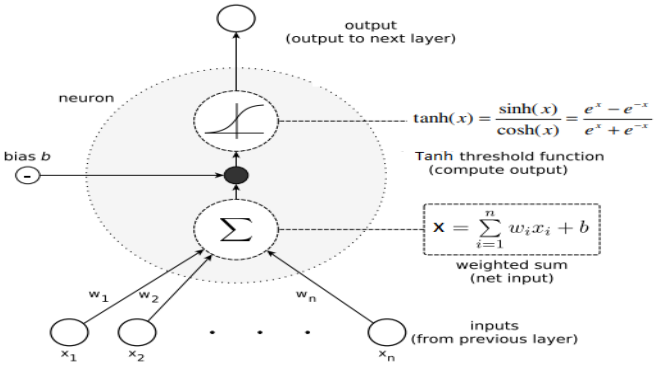


Figure 2.8: The Tanh threshold unit is designed to represent non-linear functions. hyperbolic tangent function offers better recognition accuracy than any other functions.

2.3.3.6 Feed-Forward Neural Networks and Backpropagation

In feed-forward neural networks (FFNNs), ensembles of neurons are arranged in layers, with each neuron computing a weighted sum of its inputs. The input neurons get signals coming from the environment and the output neurons provide signals to the environment. Those neurons that are not directly connected to the environment, but which are interconnected with other neurons, are known as hidden neurons.
Feedforward neural networks are closed-loop and fully connected. Meaning that each neuron supplies an input to each neuron of the next layer.
The most widely known type of neural network training technique is the error backpropagation algorithm. It applies gradient descent to learn the weights of multilayer networks. multilayer. It operates in small and iterative steps, moving from the output layer to the input layer. The activation function of the neuron has to be differentiable(Staudemeyer et al. 2019).

2.3.3.7 Recurrent Neural Networks

Recurrent neural networks (RNNs) are very dynamic systems; they have an inner state at every time step. This is achieved through circular interconnections between neurons in the top and bottom layers as well as through optional self-feedback connections. Such feedback connections permit RNNs to propagate data from earlier events to current steps of processing. In this manner, RNNs build a memory of time series events.
RNNs range from the partially to the fully connected ,Elman 1990 suggested a similar three-layer neural network, but in addition, the outputs of the hidden layer are saved in cells known as ” contextual cells ”. The output of a context cell is re-injected in a circular loop into the hidden neuron alongside the original signal. Each hidden neuron owns a context cell and gets input both from the input layer and from its context cells. Elman networks can be trained by standard error backpropagation, with the output of the context cells simply considered as additional data. The output of the context cells is simply considered as an additional input

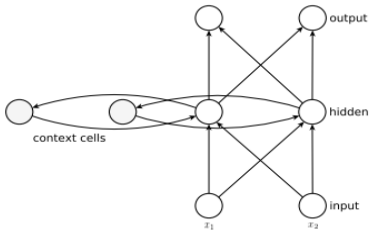


Figure 2.9: Elman neural network(Staudemeyer et al. 2019).

2.3.3.8 LSTM Cells and Their Variants

In RNNs, recurrent layers or hidden layers are made up of recurrent cells which states are both affected by past states and current input with feedforward connections. Recurrent layers can be arranged in multiple architectures to make a variety of RNNs. As a result, RNNs are characterized primarily by the recurrent cell and the network architecture .

2.3.3.9 An original LSTM: No Forget Gate

The original paper introducing the standard LSTM cell was published in 1997(Hochreiter et al. 1997). A simple RNN cell was enhanced by appending a memory block that is controlled by input and output multiplicative gates.

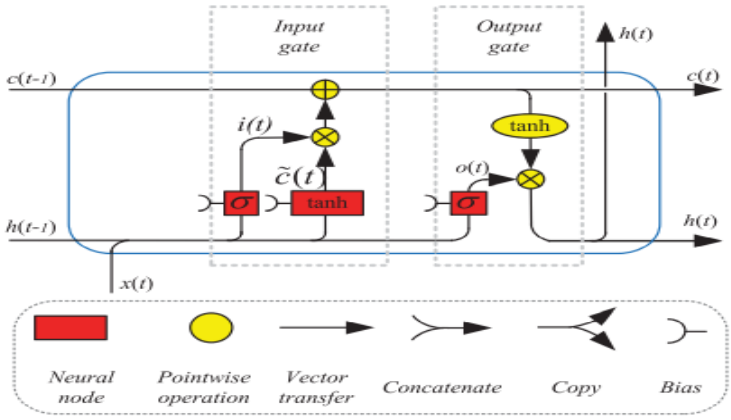


Figure 2.10: The Original LSTM architecture(Yu et al. 2019).

The input gate is capable of blocking irrelevant data from being stored in the cell state, and the output gate decides what information can be output based on the cell state(Yu et al. 2019).

2.3.3.10 LSTM with a Forget Gate

gers2000learning Have modified the initial LSTM in 2000 by implementing a forgetting

gate in the cell,The forget gate can determine which information will be dropped from the cell's state.

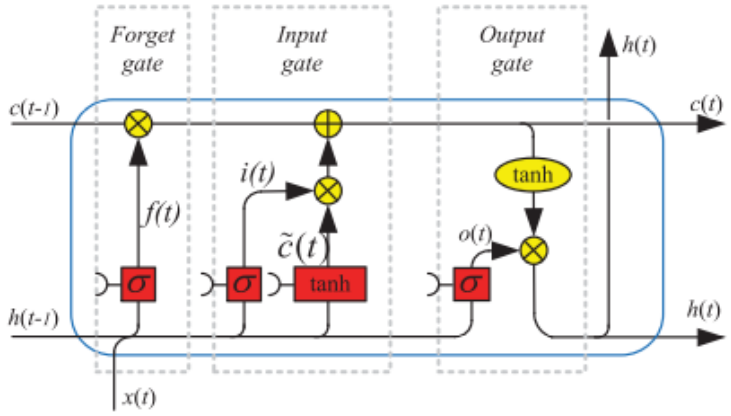


Figure 2.11: The LSTM architecture with a forgetting gate(Yu et al. 2019).

2.3.3.11 LSTM architecture with a peephole connection

As the gates of the preceding LSTM cells have no direct links to the cell state, there is a loss of essential information which affects the network performance. In order to solve this problem, Gers et al. 2000 enhanced the LSTM cell by bringing in a "peephole" connection

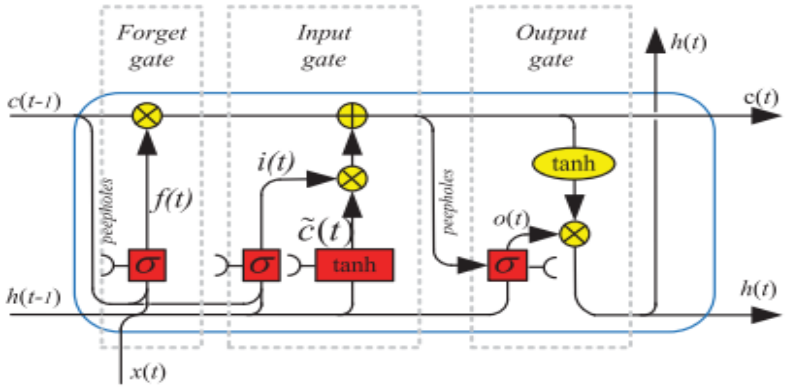


Figure 2.12: LSTM architecture with a peephole connection(Yu et al. 2019).

2.3.3.12 Bidirectional recurrent neural networks

The fundamental concept of bidirectional recurrent neural networks (BRNNs) is to introduce each sequence of forward and backward learning into two separate recurrent networks, both of them connected into the same output layer(Sønderby et al. 2014). In the classical recurrent neural network model and LSTM model, only forward propagation

of information is supported, so the state at time t only depends on the given information preceding time t. In an effort to make sure that each instant contains the context information, the BiLSTM model, combining bidirectional recurrent neural network (BiRNN) models and LSTM based units, is implemented to capture the contextual information(Xu et al. 2019)

2.3.4 An Introduction to Convolutional Neural Networks

Deep convolutional neural network (CNN) is a particular class of neural networks, which has demonstrated excellent results in various computer vision and image processing related challenges. Some of the exciting application areas of the CNN are image classification and segmentation, object detection, video processing, natural language processing and speech recognition. The powerful learning capacity of deep CNN is principally due to the use of multiple feature extraction stages that can automatically learn features from the data,The CNN topology itself consists of multiple learning levels that are composed of a combination of convolutional layers, nonlinear processing units, and subsampling layers. The CNN is a multilayer feedforward network, in which each layer, using a set of convolutional kernels, makes multiple transformations. The convolution process provides useful feature extraction from the locally correlated group of data items(Khan et al. 2020). The ability of CNNs to learn and then automatically extract features from raw input data can be extended to time series forecasting problems. A sequence of data observations can be handled as a one dimensional image that a CNN model can interpret and distill to its most significant features(Brownlee 2018).

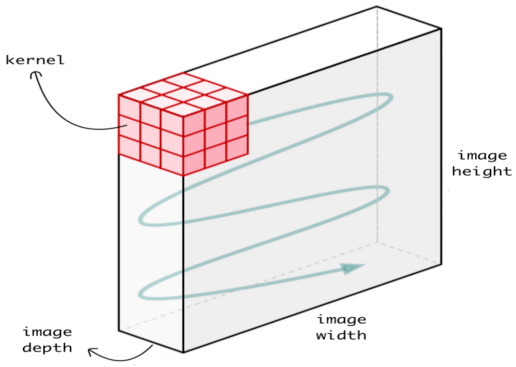


Figure 2.13: Kernel dragging over data and extracting features

2.3.5 Basic CNN components

2.3.5.1 Convolutional Layer

In the convolutional layer, several filters slide over the layer for the input data set. A summation of an element-by-element multiplication of the kernel and the input field size is then calculated as the output of the layer. The weighted sum is defined as the input element of the following layer(Albawi et al. 2017).

Convolutions have traditionally been used to process images, but they can also be applied to perform other operations, CNNs enforce a local connectivity pattern between neurons of adjacent layers.The first layer of a convolutional neural network will always be a convolutional layer. Convolutional layers perform a convolution operation on the input and forward the results to the next layer. A convolution transforms all its receiving field pixels into one single value. For example, if you are going to apply a convolution to an image, you are reducing the frame size and collecting all the information in the field into one single pixel. The output of the convolutional layer is a vector. According to the type of task we must perform and the kind of properties we are looking to acquire, we can use different types of convolutions.

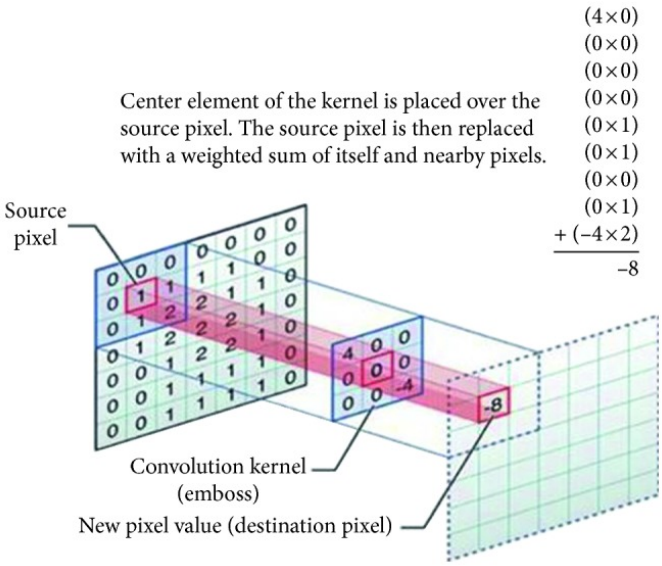


Figure 2.14: Convolution layer slides the filter over a given input. Output is the summation of an element by the element matrix multiplication of the filter and receptive field (Albawi et al. 2017)

2.3.5.2 Pooling layer

The goal of the pooling layers is to reduce progressively the dimensionality of the representation, and so to reduce further the number of parameters and the model's computational complexity. The pooling layer is operating on each input activation map, and is scaling its own size(O'Shea et al. 2015). The the most popular pooling mode, max pooling, represents the maximum value inside the pooling filter, There are other pooling techniques, such as averaging and summation.

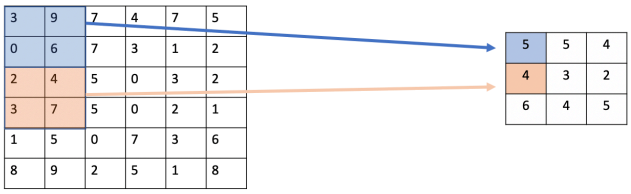


Figure 2.15: Example of (2,2) max pooling

2.3.5.3 Fully-connected layer

The fully connected layer has neurons which are directly connected to the neurons in adjacent layers, but are not connected to any of the layers within them. This is similar to the way neurons are arranged in traditional forms of ANN(O'Shea et al. 2015).

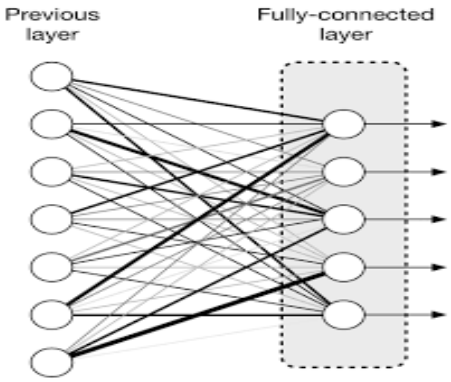


Figure 2.16: Fully-connected layer

2.3.6 Conv-LSTM Networks for time series prediction

The CNN is typically applied to get the spatial properties, while the LSTM is implemented to extract the temporal pattern. Although some studies propose hybrid deep learning methods by combining multiple models to capture various features, such as spatial, temporal and periodic,Conv-LSTM is more capable of catching spatio-temporal characteristics because it treats spatial and temporal features as a combined unit, which enhances the forecast performance(Zheng et al. 2020)

2.3.7 TimeDistributed layer

[1] Using of this wrapper, a layer can be attached to every time slot of the input. Every single input has to be at least 3D, and the dimension of the first input's index one will be taken as the time dimension.

[1] https://keras.io/api/layers/recurrent$_l ayers/time_d istributed/$

2.4 Conclusion

Nowadays, more data means more performance, but the classical statistical method cannot deal with time series problems, so we worked on the hybridization of deep learning methods which will be the best solution for our problem and will allow us to extract the spatiotemporal constraints in order to use them later on to make high accuracy predictions.

Chapter 3

Conception of the proposed system

3.1 Introduction

In the previous chapters we presented generalities,architecture and all necessary explanations on IoT and the AI.In this chapter we will talk about the system requirements, description of the softwares in addition to the hardware needed to carry out this project.

3.2 System requirements analysis

3.2.1 Functional requirements

Those are the requirements needed for the system to be fully functional:

-Implementing and configuring an MQTT service(broker, publisher and client).

-Acquire measurements of physical quantities from the sensors used.

-Being able to read from a serial monitor.

-Storing data in a My-SQL database.

-Monitoring of data using an IoT platform.

-Data collection, analysis and pre-processing to ensure the performance of our Forecasting Model.

-Build a deep learning model and a SARIMA model capable of performing air quality prediction.

3.2.2 Non-Functional requirements

In what follows are the non-functional requirements needed to be considered in our project:

-Data security.

-Compatibility with any operating system.

-Ergonomics of graphic interfaces.

-Scalability.

3.2.3 Vision of the proposed solution

The solution aim to real-time data collection, monitoring, storing and prediction with indoor air quality types of sensors such as particulates Matter (PM2.5,PM10,PM100), TVOC, CO2, Temperature, etc.

The solution to offer is based on the IoT systems paradigm.To carry out this type of work, its necessary to elaborate a specifications document that includes all the necessary hardware and software to achieve this project.

3.2.3.1 Hardware needs

After doing the necessary researchs the following are the principal hardware needs in our project:

-Raspberry Pi3 B+ board.

-ESP32 board.

-Particulates sensor(PMSA003i).

-Temperature, pressure, humidity, gas resistance and altitude sensor (BME680).

-CO2,TVOC sensor(CCS811).

-Wires,LEDs,bread board.

-18 Watts power supply.

-Powerful computer .

3.2.3.2 Software needs

During the carriying out of this project we used the following software which were easy to use and very helpful:

-Visual Studio Code.

-Mosquitto Open-source broker.

-NODE RED.

-Platform IO.

-Docker.

-Github.

-My-SQL.

-Python 3.9

-Jupyter notebook

3.2.4 How does the system work?

Starting with data collecting, the sensors are connected to an esp32 board using the SPI protocol for the CCS811 sensor, and the two other (BME680,PMSA0003i) sensors with the I2C protocol.

In our work the raspberry will play a major role in networking.To facilitate the task and make it more manageable we installed a Docker on it and after that we installed differents images on it to make containers for each of mosquitto broker,node js (for EXPRESS js),NODE RED and My-SQL.

The collected data is sent via the integrated Wi-Fi in the esp32 board to differents topics using the MQTT protocol to the mosquitto (MQTT Broker).Once data is sent to the broker, the client Express JS is connected to the same network along with the broker

and subscribed to the same topics, as where the data is sent will get access to the data and it will get stored to the Database, and also the NODE RED client subscribed to the same topic as where the data is sent will have the ability to be monitored as it will be shown in the tests part.

In order to perform a prediction, the data we have collected must be first normalized and partitioned into three channels: the first one contains the difference between each day and the previous day to give our model an overview of the trend, the second one contains the difference between the current value and the same value compared to that day of the previous year and the last and main one contains the current value which will be combined with the second one to give our model an overview of the seasonal pattern; after all this, our data will be reshaped into a 12-day time window. thus, with the format of 12 days and three channels, we are going to be able to make a good prediction of the coming day's PM25 concentrations. Moreover, with the average value of this concentration in this month, we can estimate the next month's average value with the SARIMA model.

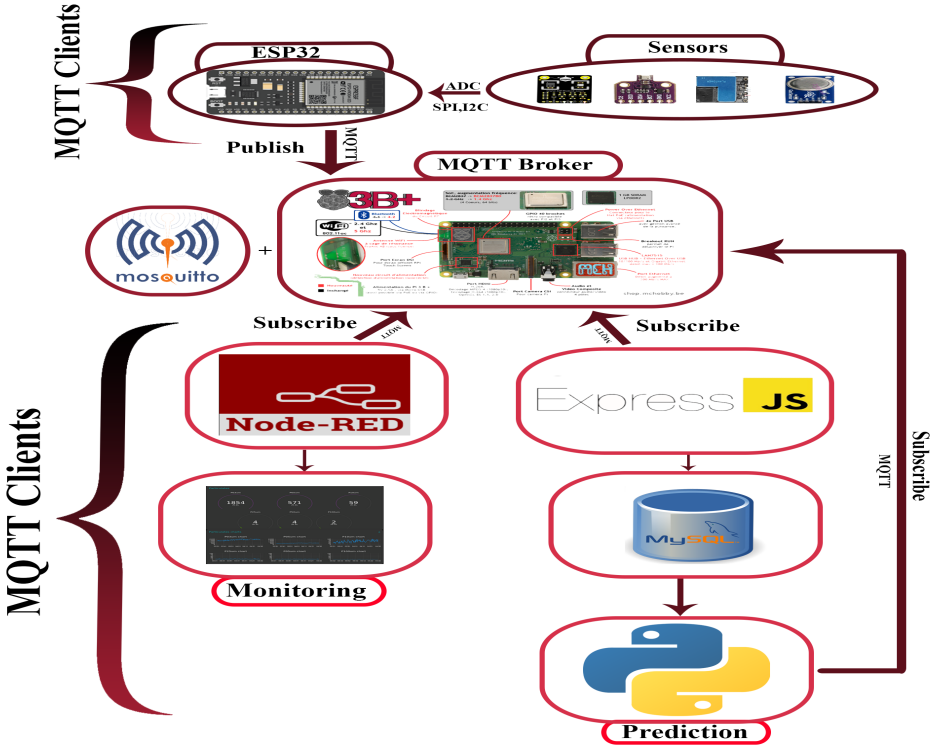


Figure 3.1: The IoT System's Architecture

And now after we have described the system in general and studied the specifications we can now move on to the second part where we are going to define and presents the Hardware and Software architecture of materials that have been used.

3.3 Hardware description

3.3.1 Raspberry PI3

3.3.1.1 Defintion

The Raspberry Pi is a credit card-sized, single-board ARM nanocomputer designed by professors at the University of Cambridge's Computer Science Department as part of the Raspberry Pi Foundation[1]. The Raspberry Pi was created to democratize access to computers and digital making. This democratization is possible because of the low cost of the Raspberry Pi, but also thanks to free software[2]. The Raspberry Pi supports several variants of the free GNU/Linux operating system, including Debian, and other compatible software.

The raspberry pi 3's specs are available in appendix D.1

3.3.2 Esp32

3.3.2.1 Definition

The ESP32 is a series of low-cost, low-power system-on-chip microcontrollers with integrated dual-mode Wi-Fi and Bluetooth. The ESP32 series uses either a Tensilica Xtensa LX6 microprocessor in dual-core and single-core versions, a dual-core Xtensa LX7 microprocessor, or a single-core RISC-V microprocessor, and includes integrated antenna switches, an RF balun, a power amplifier, a low-noise receive amplifier, filters, and power management modules. The ESP32 is created and developed by Espressif Systems, a Chinese company based in Shanghai, and is manufactured by TSMC using their 40nm[3]. process. It is the successor to the ESP8266 microcontroller.

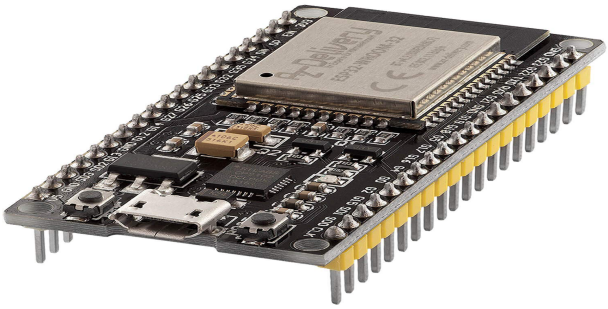


Figure 3.2: ESP32 board

The features of the ESP32 are available in appendix D.2.[4]

[1] https://www.raspberrypi.org/,Mai 2022.
[2] https://www.raspberrypi.org/,Mai 2022.
[3] https://www.espressif.com/en/products/socs/esp32,Mai 2022.
[4] https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf,Mai 2022.

3.3.3 sensors

3.3.3.1 PMSA00I

[5]This sensor is great for monitoring air quality, in a compact plug-in format and what makes it different from the others is that this one has i2c interface.

This sensor uses laser scattering to radiate suspending particles in the air, then collects scattering light to obtain the curve of scattering light change with time. The microprocessor calculates equivalent particle diameter and the number of particles with different diameters per unit volume.

The I2C data stream updates once per second, we will get:

-PM1.0, PM2.5 and PM10.0 concentration in both standard environmental units.

-Particulate matter per 0.1L air, categorized into 0.3um, 0.5um, 1.0um, 2.5um, 5.0um and 10um size bins.

The features of the sensor are available in appendix D.3.



Figure 3.3: PMSA003I sensor

3.3.3.2 SEN0035

[6]Based on the combination of CCS811+BME280 chip, this module features high accuracy, IIC interface and fast Measurement. The BME280 can provide temperature and humidity compensation for CCS811 to improve the whole accuracy to a certain extent. It can be used to detect temperature, humidity, barometric pressure, altitude, TVOC and eCO2. CCS811 air quality sensor uses AMS's unique micro-hot plate technology. Compared with conventional gas sensors, it has lower power consumption, shorter preheating time, and smaller size. The internally integrated ADC and MCU allow it to collect and process data, and return via I2C.

BME280 is an environmental sensor that combines temperature sensor, humidity sensor and barometer in one board. It has high precision, multiple functions, small size, etc. The sensor offers ±0.5℃ temperature error and ±2%RH humidity error. It provides very stable performance within the detection temperature range. Besides, the offset

[5]https://cdn-learn.adafruit.com/downloads/pdf/pmsa003i.pdf,April 2022.
[6]https://cdn-learn.adafruit.com/downloads/pdf/pmsa003i.pdf,April 2022.

temperature coefficient is ±1.5 Pa/K, equiv. to ±12.6 cm at 1 °C temperature change.

The following table 3.1 shows the effects of carbon dioxide and TVOC on the human body.

Carbon Dioxide (PPM)	Effect on Human	TVOC Concentration (PPB)	Effect on Human
<500	Normal	<50	Normal
500-1000	A little uncomfortable	50-750	Anxious,uncomfortable
1000-2500	Tired	750-6000	depressive, headache
2500-5000	Unhealthy	>6000	headache and other nerve problems

Table 3.1: The effects of carbon dioxide and TVOC on the human body

The specifications are available in appendix D.4



Figure 3.4: SEN0335 sensor

3.3.3.3 BME680

[7]The BME680 is the first gas sensor that integrates high-linearity and high-accuracy gas, pressure, humidity and temperature sensors. It is especially developed for mobile applications and wearables where size and low power consumption are critical requirements. The BME680 guarantees - depending on the specific operating mode - optimized consumption, long-term stability and high EMC robustness. In order to measure air quality for personal wellbeing the gas sensor within the BME680 can detect a broad range of gases such as volatile organic compounds (VOC).
Specefication are availabale in appendix D.5.

[7]https://www.bosch-sensortec.com/products/environmental-sensors/gas-sensors/bme680/#:~:text=The%20BME680%20is%20the%20first,power%20consumption%20are%20critical%20requirements.,Mai 2022.

3.3.3.4 MQ135

[8]Air quality sensor for detecting a wide range of gases, including NH3, NOx, alcohol, benzene, smoke and CO2. Ideal for use in office or factory. MQ135 gas sensor has high sensitivity to Ammonia, Sulfide and Benze steam, also sensitive to smoke and other harmful gases. It is with low cost and particularly suitable for Air quality monitoring application.
Specification are available in appendix D.6.

3.3.4 Electronic schematics

In Fig 3.5 is a description of the wiring for our electronic circuit.

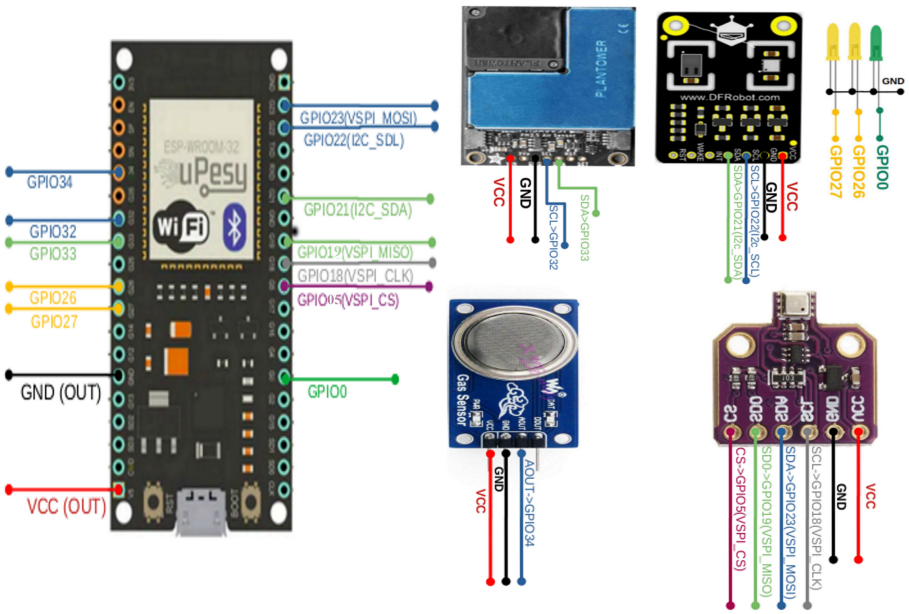


Figure 3.5: Electronic Schematics of the edge device

The sensors are connected to an esp32 board using the SPI protocol for the CCS811 sensor, and the two other (BME680,PMSA0003i) sensors with the I2C protocol, the power source for the sensors is supplied from the ESP32.

[8]https://www.rhydolabz.com/sensors-gas-sensors-c-137_140/air-quality-sensor-mq135-p-1115.html#:~:text=Description%3A%20Air%20quality%20sensor%20for,smoke%20and%20other%20harmful%20gases.,Mai 2022.

3.4 Software description

3.4.1 Visual Studio Code

Visual Studio Code is an extensible code editor developed by Microsoft for Windows, Linux and macOS. we need to chose the right. Features include debugging support, syntax highlighting, smart code completion, snippets, code refactoring, and integrated Git. Users can change the theme, keyboard shortcuts, preferences and install extensions that add additional functionality.

The source code for Visual Studio Code comes from Microsoft's free and open source software project VS Code released under the permissive MIT license, but the compiled binaries are freeware, which is free for any use but proprietary.

It's interface shown in Fig C.5 make using it so easy and very helpful for developing in any programming language.

3.4.2 Platform io

PlatformIO is a cross-platform, cross-architecture, multi-framework professional IDE tool for embedded system and software engineers who write embedded applications. By providing a universal IDE interface using PlatformIO, you are able to program your hardware in a more developer-friendly way!

In this project we worked with Platform io as an extension in VS Code and its interface is shown in Fig C.6.

3.4.3 mosquitto

Mosquitto is the most used broker for asynchronous programming in the IoT and Industrial IoT (IIoT) projects. Launched in 2008, it is available on all platforms (MacOS, Windows XP-10, Linux).And for that we are using it during this project.

3.4.4 Express JS

In our application we used as a MQTT client in order to connect it to the data base and storing data in it.

3.4.5 My-SQL

MySQL is a relational database management system (RDBMS) developed by Oracle that is based on structured query language (SQL).

Often called the most popular database and currently enjoying widespread, effective use regardless of industry, it's clear that anyone involved with enterprise data or general

IT should at least aim for a basic familiarity of MySQL[9].

3.4.6 NODE RED

Node-RED is a flow-based programming tool, originally developed by IBM's Emerging Technology Services team and now a part of the OpenJS Foundation.Node-RED consists of a Node.js based runtime that you point a web browser at to access the flow editor. Within the browser you create your application by dragging nodes from your palette into a workspace and start to wire them together. With a single click, the application is deployed back to the runtime where it is run.

The palette of nodes can be easily extended by installing new nodes created by the community and the flows you create can be easily shared as JSON files.[10].

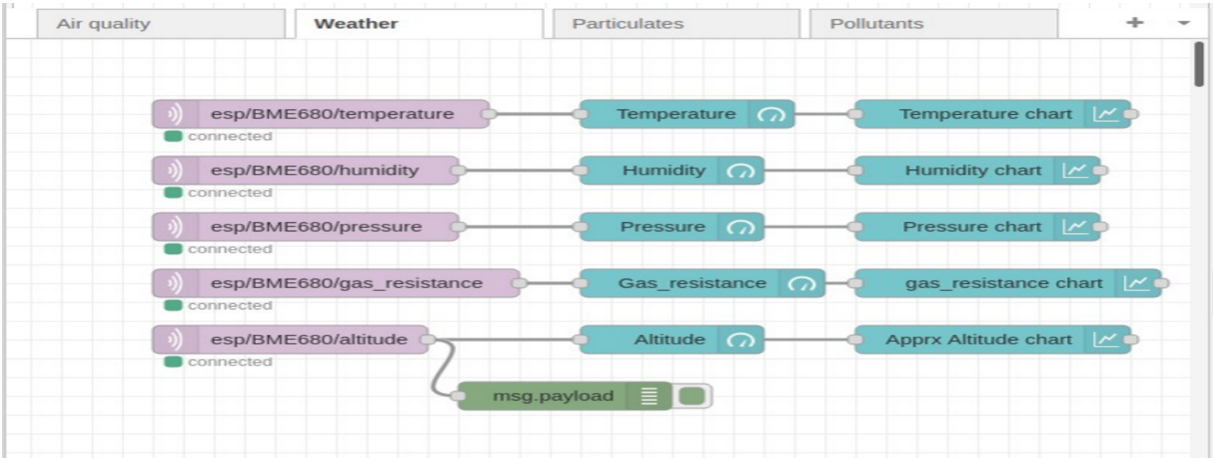


Figure 3.6: A NODE RED flow example

And here is a script formatted example of a NODE RED flow:

```
[
 {
        "id": "7d9f66928278cbf6",
        "type": "tab",
        "label": "Pollutants",
        "disabled": false,
        "info": "",
        "env": []
    }
]
```

[9] https://www.talend.com/resources/what-is-mysql/,April 2022.
[10] https://nodered.org/about/,April 2022.

3.4.7 Github

At a high level, GitHub is a website and cloud service that helps developers store and manage their code, as well as track and monitor changes to it. To understand exactly what GitHub is, we need to know two related principles:

3.4.7.1 Version control

Version control helps developers track and manage changes to the code in a software project. As a software project grows, version control becomes essential.

Also Version control allows developers to work safely through branches and merges. With branching, a developer duplicates a portion of the source code (called the repository). The developer can then safely make changes to that part of the code without affecting the rest of the project. Then, once the developer has managed to get his or her part of the code working properly, he or she can merge that code with the main source code to make it official. All these changes are then tracked and can be undone if necessary.

3.4.7.2 Git

Git is a specific open-source version control system created by Linus Torvalds in 2005. In practical terms, Git is a distributed version control system, which means that the entire code base and history is available on each developer's computer, allowing easy branching and merging.

3.4.8 Docker

[11]Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allows you to run many containers simultaneously on a given host. Containers are lightweight and contain everything needed to run the application, so you do not need to rely on what is currently installed on the host. You can easily share containers while you work, and be sure that everyone you share with gets the same container that works in the same way.

3.4.9 Python

Primarily, Python is a top-level, open source, fully interpreted programming language that provides an extremely excellent approach to object-oriented development. It is among

[11]https://docs.docker.com/get-started/overview/,Mai 2022.

the best languages that are used by data scientists for a variety of data science applications/projects. There are great features in Python to handle mathematics, statistics, and scientific functions. It provides great libraries to handle data science applications such as sklearn , Tensorflow , statsmodels , Librosa and others...

3.4.10 Jupyter Notebook

[12] Jupyter is the premier interactive web-based development framework for notebooks, coding, and data. It offers a highly flexible interface allowing Users to setup and manage data science, scientific computing, computational journalism, and machine learning workflows.

3.5 Conclusion

Presenting the conception behind our work is preliminary in order to move to the next chapter.In this chapter we presented the technologies, tools and a detailed architecture of our system from hardware to software side which will allow us to successfully do the implementation and show the results in the next chapter.

[12] https://jupyter.org/,Mai 2022.

Chapter 4

Realisation and Results

4.1 Introduction

After discussing and explaining the system's conception in the previous chapter we will now do the implementation and show the results obtained after that.

In this chapter we show the several installations and configurations made in order to make our system work and also provided a link to a Github repository in order to explore the source code.

4.2 Raspberry Pi3

4.2.1 Installing the operating system

In order to install Raspberry Pi OS Lite (Legacy) operating system we used the Raspberry pi imager software as shown in fig 4.1

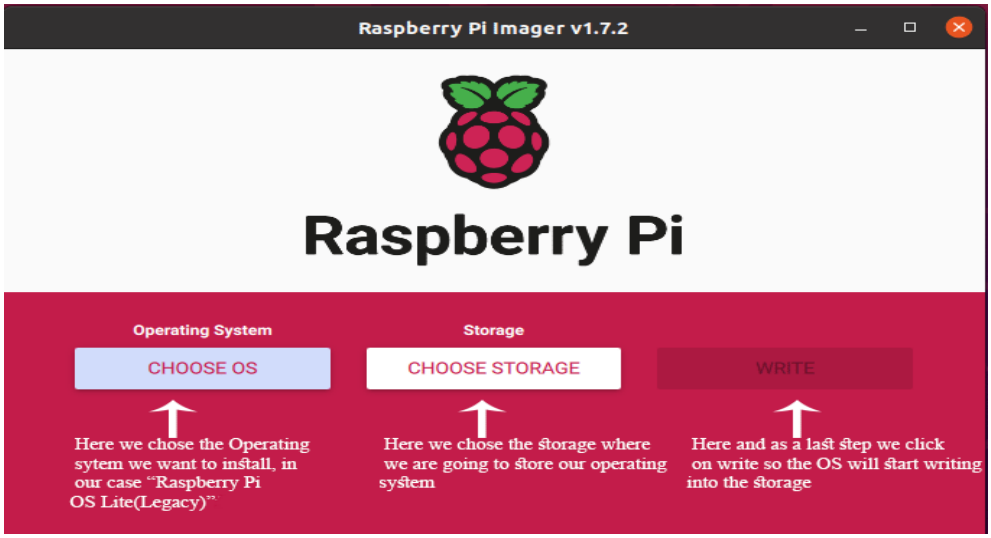


Figure 4.1: Raspberry Pi imager

4.2.2 Installing Git

We installed git in order to keep our work updated and pull changes if there's any.

To install git in our raspberry pi3 we need to run the following commands in the terminal:

```
$ sudo apt update
$ sudo apt install git
```

Now to check if our system supports git, we run the following command:

```
$ git --version
```

4.2.3 installing Docker

The first step is to update packages for any changes then install docker using the following commands:

```
$ sudo apt-get update
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

In our work we worked with docker compose in order to facilitate the work, so we need also to install docker compose using the following command:

```
$ apt-get install docker-compose
```

Now to check the docker and docker compose version we run the following commands:

```
$ docker --version
$ docker-compose --version
```

Now we create a docker-compose.yml file and configure the differents containers as shown in the following script:

```yaml
version: '3'
services:
  node-app:
    build: .
    network_mode: host
    ports:
      - "3000:3000"
    environment:
      - PORT=3000
    depends_on:
      - database
    restart: always
  database:
    image: hypriot/rpi-mysql
    restart: always
    network_mode: host
    ports:
      - "1883:1883"
      - "9001:9001"
```

```
    environment:
      - MYSQL_ROOT_PASSWORD=Cherfianadir2022@
      - MYSQL_DATABASE=iot_air_quality
    volumes:
      - mysql-db:/var/lib/mysql
      - ./:/var/lib/mysql
  mqtt:
    image: mjenz/rpi-mosquitto
    network_mode: host
    restart: always
  node-red:
    image: nodered/node-red
    restart: always
    network_mode: host
    volumes:
      - node_red_data:/data
    depends_on:
      - mqtt
volumes:
  mysql-db:
  node_red_data:
```

In order to make the work more organised we create two more other docker-compose-development.yml and docker-compose-production.yml, one for developemnt process and the second for production:

Development

```
version: '3'
services:
  node-app:
    build:
      context: .
      args:
        NODE_ENV: development
    volumes:
      - ./:/app:ro
      - /app/node_modules
    environment:
      - NODE_ENV=development
    command: npm run dev
```

Production

```
version: '3'
services:
```

```
  node-app:
    build:
      context: .
      args:
        NODE_ENV: production
    environment:
      - NODE_ENV=production
    command: node app.js
```

Now we clone the directory that we have already created from our github repository :

```
$ git clone https://github.com/NadirMohamedCHERFIA/Dockerize.git
```

Then we run the following commands in order to build our containers:

```
$ cd Dockerize
$ docker-compose COMPOSE_HTTP_TIMEOUT=3600 -f docker-compose.yml -f
docker-compose.prod.yml up -d --build
```

Now we check the running docker in our system:

```
$ docker ps
```

Results

```
pi@raspberrypi:~ $ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS               NAMES
1514ba841879        dockerize_node-app  "docker-entrypoint.s…"   5 weeks ago         Up 3 hours                              dockerize_no
de-app_1
16c944abb97f        nodered/node-red    "npm --no-update-not…"   5 weeks ago         Up 3 hours (healthy)                    dockerize_no
de-red_1
90c9a6e7fe50        hypriot/rpi-mysql   "/entrypoint.sh mysq…"   5 weeks ago         Up 3 hours                              dockerize_da
tabase_1
6b8ddaed2e3c        mjenz/rpi-mosquitto "/usr/sbin/mosquitto"    5 weeks ago         Up 3 hours                              dockerize_mq
tt_1
```

Figure 4.2: Containers running in our docker

The fig 4.3 shows the docker architecture that is being used:

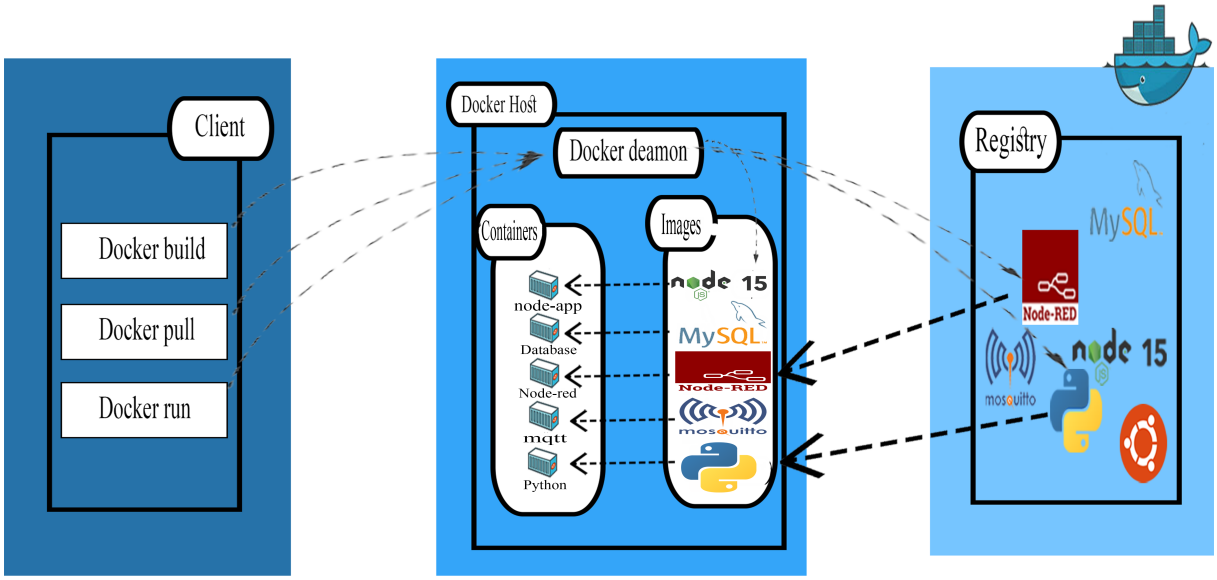


Figure 4.3: Docker architecture of our system

4.2.3.1 Configuring mosquitto

1-Getting into mosqquitto interactive mode

```
sudo docker exec -it dockerize_mqtt_1 bash
```

2-Securing access to our mosquitto broker

```
sudo nano /etc/mosquitto/conf.d/default.conf
```

And we add the following lines:

```
/************************************/
^^I  allow_anonymous true
^^I  password_file /etc/mosquitto/pwfile
^^I  listener 1883
^^I/************************************/
```

3-Now we create our username name and its password

```
sudo mosquitto_passwd -c /etc/mosquitto/pwfile <here we puted our username>
password : <Our password>
Reenter your password :<Our passsword once again>
```

4-Now we just have to restart mosquitto

```
sudo /etc/init.d/mosquitto restart
```

4.2.3.2 Configuring mysql

1-Getting into mysql interactive mode

```
sudo docker exec -it dockerize_database_1 bash
```

2-Creating a Database

```
CREATE DATABASE iot_air_quality;
```

3-Creating a table

```
CREATE TABLE air_quality(id INT AUTO_INCREMENT PRIMARY KEY ,TEMPERATURE_C VARCHAR(20),
HUMIDITY_percentage VARCHAR(20),ALTITUDE_m VARCHAR(20),PRESSURE_hPa VARCHAR(20),PM10
VARCHAR(20),PM25 VARCHAR(20),PM100 VARCHAR(20),P03um_perdecilitre VARCHAR(20),
P05um_perdecilitre VARCHAR(20),P10um_perdecilitre VARCHAR(20),P25um_perdecilitre
VARCHAR(20),P50um_perdecilitre VARCHAR(20) ,P100um_perdecilitre VARCHAR(20)
,CO2_ppm VARCHAR(20),TVOC_ppb VARCHAR(20),AIR_QUALITY_ppm VARCHAR(20),
GAS_RESISTANCE_KOhms VARCHAR(20),TIME TIMESTAMP NOTNULL DEFAULT
CURRENT_TIMESTAMP);
```

4.2.3.3 Configuring NODE RED

In order to get the NODE RED dashboard we have first to create the necessary flows as shown in fig 4.4 then configure the mqtt-in nodes as shown in fig 4.5 after that and finally just simply press deploy :

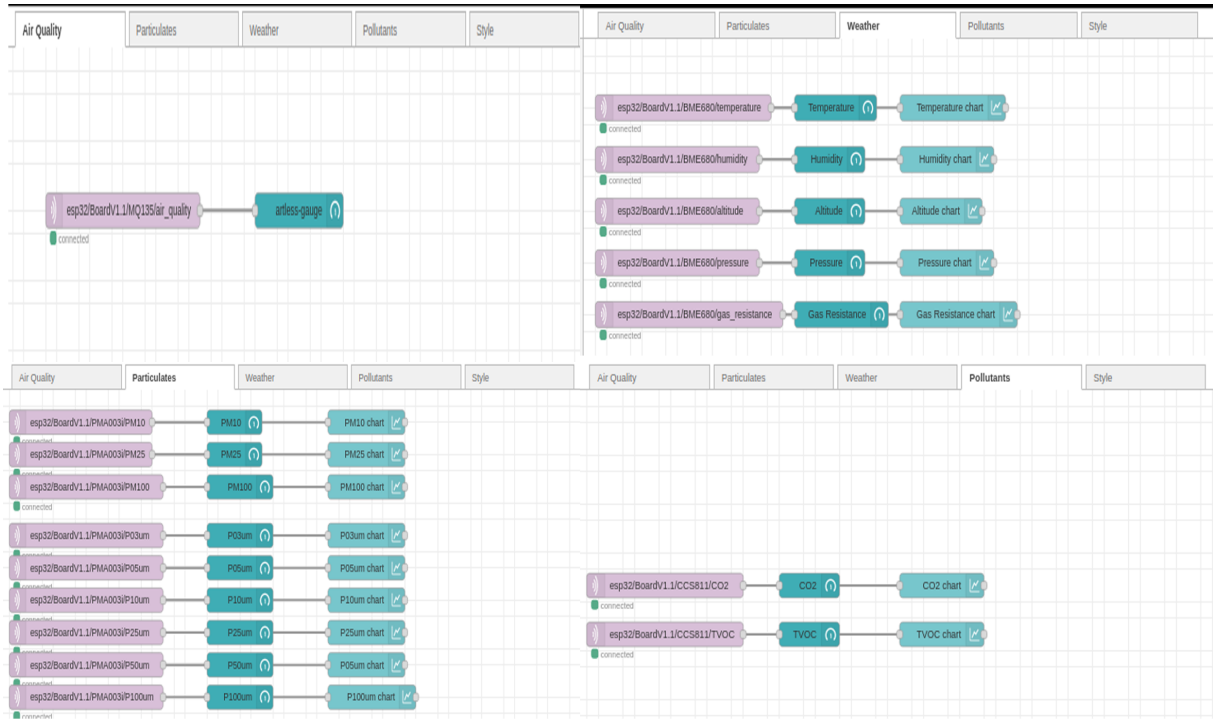


Figure 4.4: NODE RED flows creation

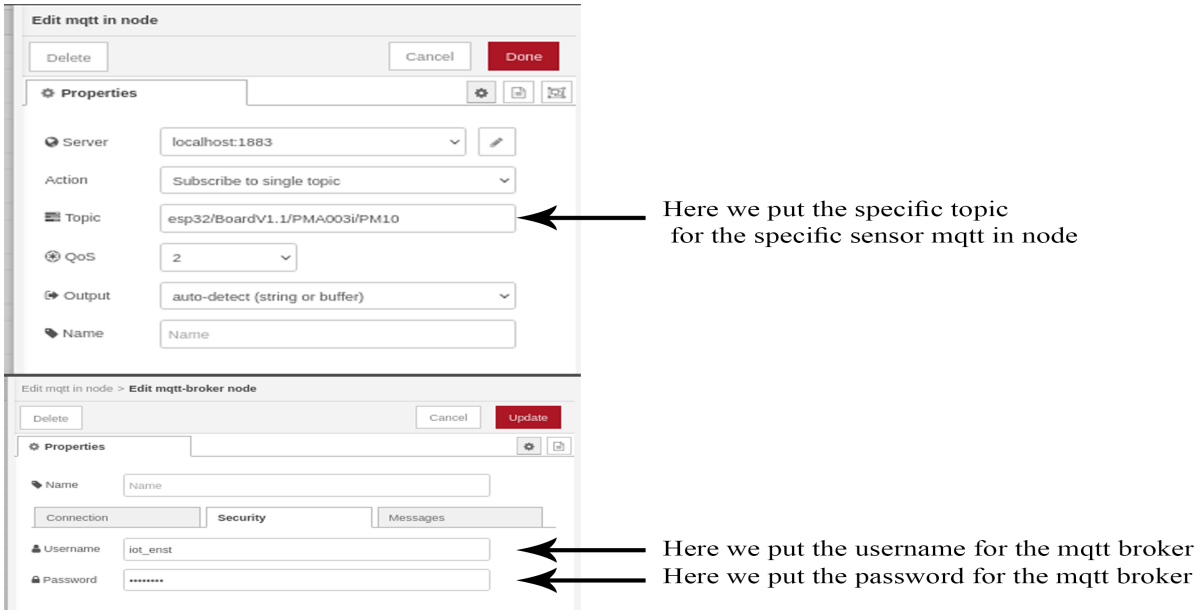


Figure 4.5: MQTT configuration in NODE RED

4.2.4 Configuring node js

In what follows are some of essentials of the Express JS used as an mqtt client while the whole code is available on Github(SE cited April 2022).

```
//Creating a connection with the data base
const db=mysql.createConnection({
    host:"localhost",
    user:"root",
    password:"Cherfianadir2022@",
    database: "iot_air_quality"
});
/*Defining the options for mqtt connection */
options={
    username:"iot_enst",
    password:"cherfianadir",
    clean:true}
    //The first client connnection
var client = mqtt.connect("mqtt://localhost",options)
client.on("connect",function(){^^I
    console.log("client1:mosquitto connected");
})
client.on("error",function(error){
    console.log("Can't connect!" + error);
    process.exit(1)});
//Defining topic
    var topic1="esp32/BoardV1.1/jsonFormatedData1";
    console.log("subscribing t/o topic");
    //Establishing connection connection with client using a QOS=1
    client.subscribe(topic1,{qos:1});
    client.on("message",function(topic1, message, packet){
        message1=message;
        console.log("message is "+ message);
        console.log("topic is "+ topic1);
    });
//Defining the listening port
const port=process.env.PORT || 3000;
app.listen(port,()=>{
    console.log(`Server started on port :\${port}`);
});
```

4.3 Edge device

In the fig 4.6 is presented the edge device algorithm architecture.

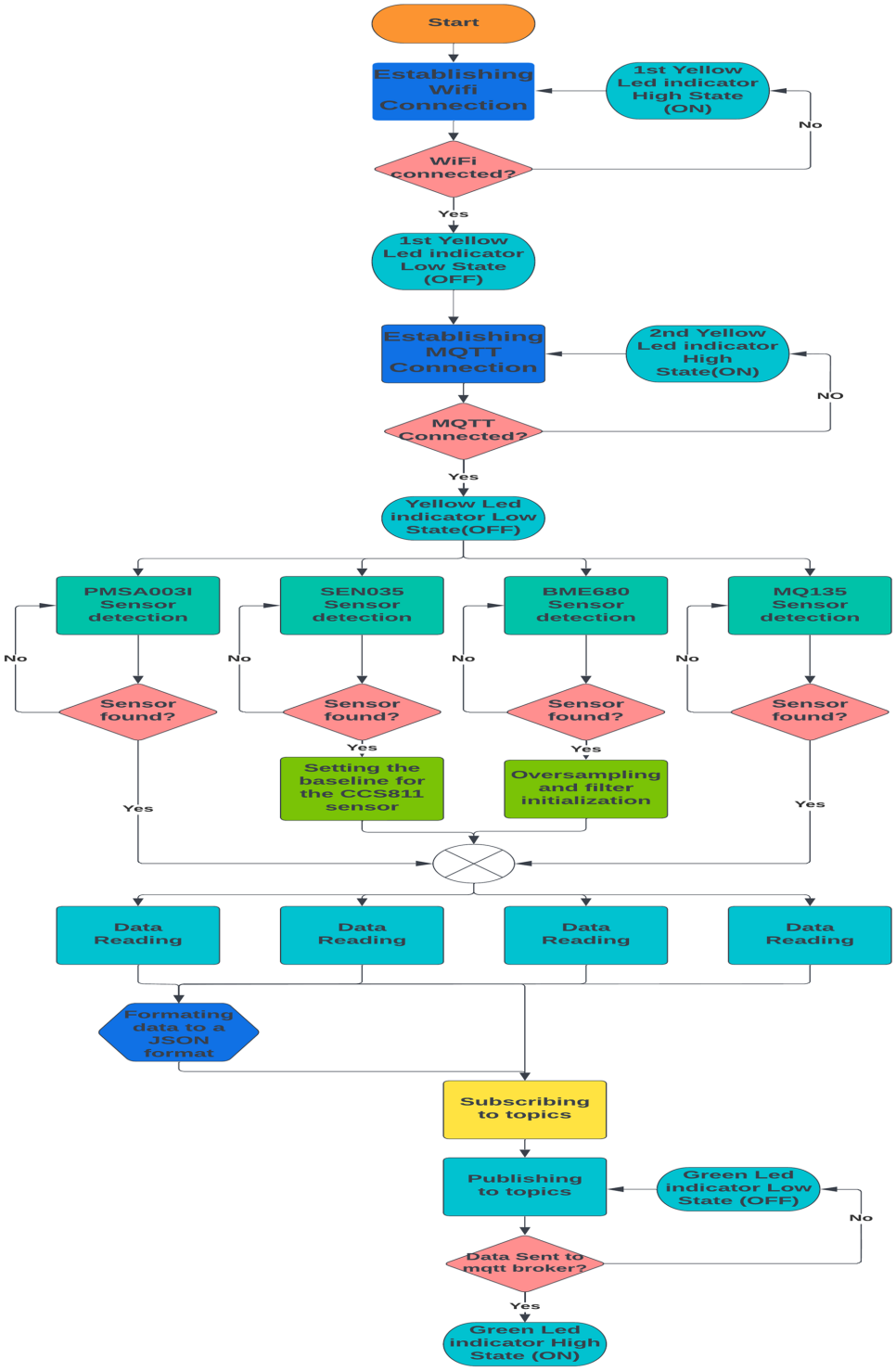


Figure 4.6: Edge device algorithm architecture

4.3.1 Prediction Model

4.3.1.1 DATA PREPROCESSING

Our training data-set (aqicn n.d.) contains 39 months of daily PM2.5 concentration records up to the present day. Therefore,This data set therefore contains many missing values due to uncontrollable reasons, so we first need to fill in the missing values through interpolation, we analyse the data in the the following table 4.1
Since time series data can exhibit a variety of patterns, it is often useful to decompose

mean	standard deviation	min	25%	50%	75%	max
68.02	15.73	32.00	58.00	64.00	74.00	172.00

Table 4.1: statistical description of PM2.5 concentration values

a time series into multiple components, each representing an underlying pattern category (Hyndman et al. 2018). In order to finalize our study, we need to decompose our time series to investigate year-to-year components such as trend, seasonality and residuals the figure 4.7 below represents the result.

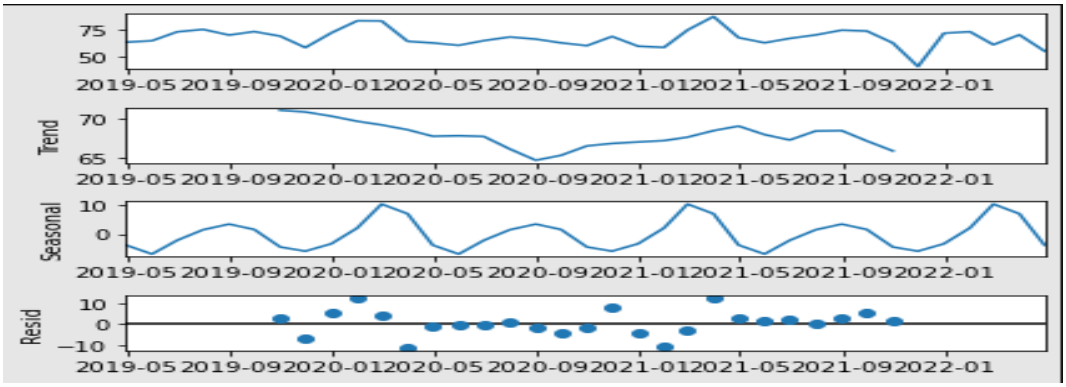
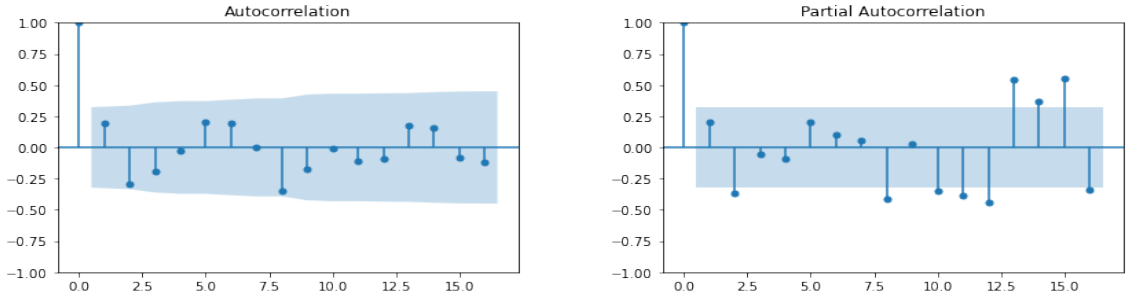


Figure 4.7: year-to-year components

. To complete our analysis, we will view the ACF and PACF plots.



(a) ACF Plot (b) PACF Plot

Figure 4.8: ACF and PACF plots

4.3.1.2 THE HYBRID CNN-LSTM MODEL

We decided to develop a hybrid CNN-LSTM model combining CNN and LSTM to improve the forecast accuracy,The CNN is typically applied to get the spatial properties, while the LSTM is implemented to extract the temporal pattern. The proposed model has PM2.5 time series as input in three channels, as already discussed, and as it is clear in the figure 4.7, there are both seasonal and trend patterns, and in the end a single time series with several steps as output.

Overfitting is a frequent phenomenon in deep neural networks (DNNs) and many solutions exist and Of all the solutions, dropout is one of the simplest and most performant. Dropout refer to the point where during the DNN training process, a cell is temporally dropped out of the network depending on a certain probability (Srivastava et al. 2014). In order to prevent over-fitting, an Exclusion Layer is introduced, the output of which is plugged into the LSTM layer for prediction and finally connected to an Fully Connected Layer , the full architecture is avaible in C.3

4.3.1.3 THE SARIMA Model

After checking the plots of acf and pacf, we can see in the figure 4.8b that there's a high partial correlation with the three preceding time lags and an yearly one with the first two months of the past year . For the acf, it only depends on the first earlier lag, as shown in the figure 4.8a.

So our SARIMA parameters will be as the following : **(3,0,1),(2, 0, 0, 12)**

4.4 Results

In the following section we are going to present the different results we have got after the realisation of this project, the circuit shown in fig 4.9 has been used to test result:

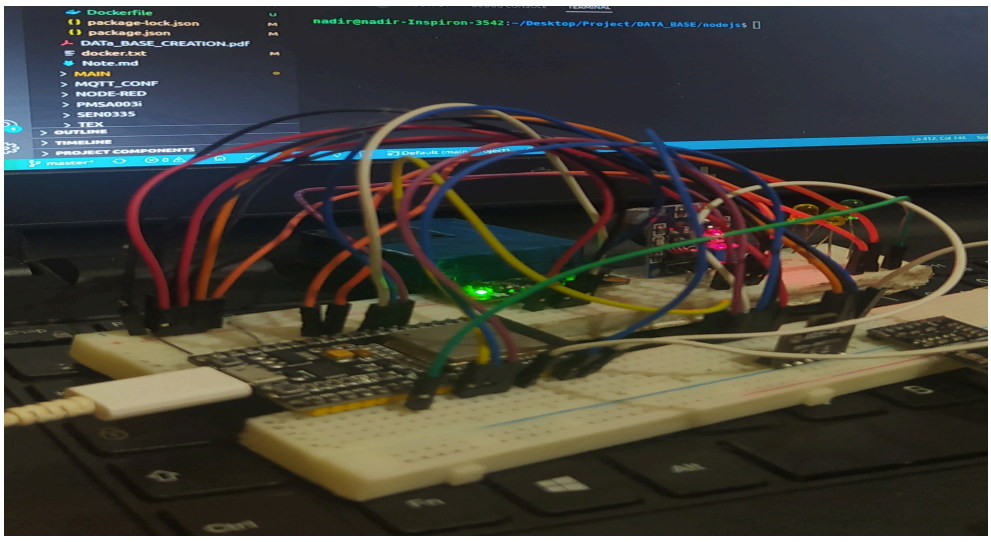


Figure 4.9: Edge device with sensors circuit

4.4.1 Data reading

After uploading the program to the ESP32 board, the serial monitor would give us something as follows:

```
*****PMA003i READING*****************************
AQI reading success
---------------------------------------
Concentration Units (standard)
---------------------------------------
PM 1.0: 11              PM 2.5: 14              PM 10: 15
Concentration Units (environmental)
---------------------------------------
PM 1.0: 11              PM 2.5: 14              PM 10: 15
---------------------------------------
Particles > 0.3um / 0.1L air:1617
Particles > 0.5um / 0.1L air:510
Particles > 1.0um / 0.1L air:67
Particles > 2.5um / 0.1L air:6
Particles > 5.0um / 0.1L air:2
Particles > 10 um / 0.1L air:0
---------------------------------------
****************BME680 READING*****************
Temperature = 31.74 *C
Pressure = 1013.50 hPa
Humidity = 50.66 %
Gas = 31.88 KOhms
Approx. Altitude = -2.16 m
---------------------------------------
*************CCS811 READING*******************
CO2: 425ppm, TVOC: 3ppb
---------------------------------------
********************MQ135***********************
AirQua=1650 PPM
---------------------------------------
{"temperature":"31.76","humidity":"50.68","altitude":"-2.16",
"pressure":"101351.00","PM10":"11.00","PM25":"14.00","PM100":
"15.00","P03um":"1617.00","P05um":"510.00","P10um":"67.00",
"P25um":"6.00"}
{"P50um":"2.00","P100um":"0.00","CO2":"408.00","TVOC":"1.00",
"AIR_QUALITY":"1650.00","GAS_RESISTANCE":"37890.00"}
Success sending first data chunk
Success sending second data chunk
```

4.4.2 Monitoring

The NODE RED dashboard gives us a monitor showing the collected data shown in fig 4.10 after deployment of nodes, the figures are of the dashboard are given in appendices C:

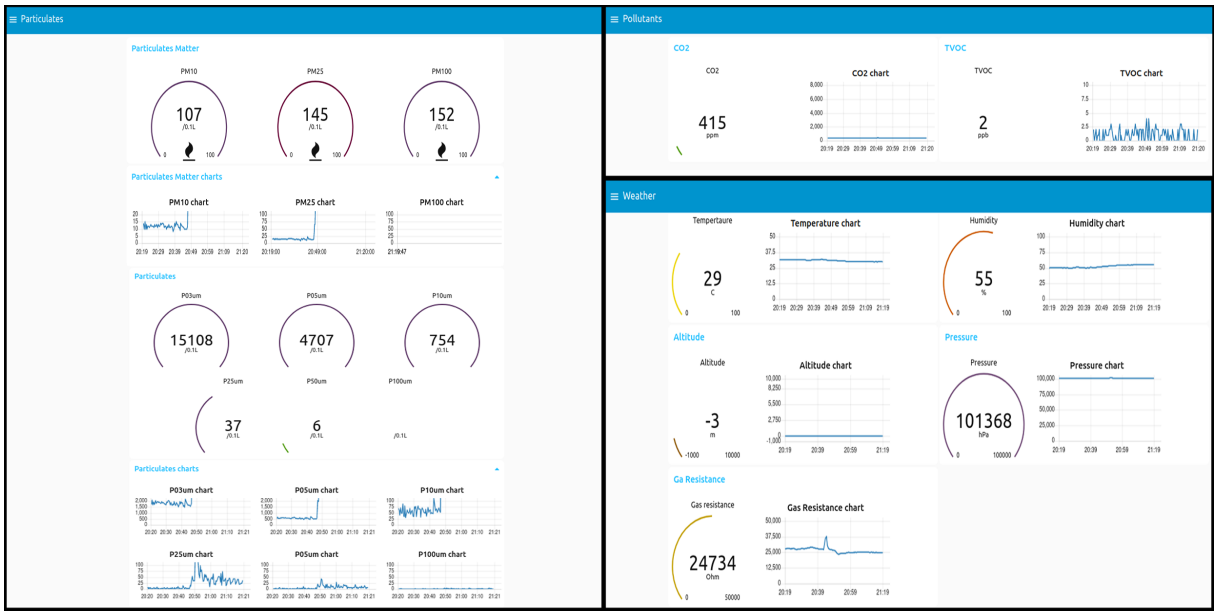


Figure 4.10: NODE RED dashboard

4.4.3 Data storing

AN example of a stored data sample is shown in fig 4.11 exported from the My-sql database:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	Id	TEMPERATURE_C	HUMIDITY_percenta	ALTITUDE_m	PRESSURE_hPa	PM10	PM25	PM100	P03um_perdecilitre	P05um_perdecilitre	P10um_perdecilitre	P25um_perdecilitre	P50um_perdecilitre	P100um_perdecilitre	CO2_ppm	TVOC_ppb	AIR_QUALITY_ppm	GAS_RESISTANCE	TIME
2	1	24.16	77.12	250.17	98356	4	4	4	804	248	13	0	0	0	400	0	1500	6391	2022-04-28 5:45:38
3	2	23.93	77.38	250.17	98356	4	6	9	786	234	47	7	5	0	417	2	1520	7847	2022-04-28 5:46:05
4	3	23.9	77.07	250.17	98356	6	6	8	921	287	18	2	2	2	411	1	1520	9536	2022-04-28 5:46:32
5	4	23.94	76.78	250	98358	3	4	5	669	203	7	2	2	0	403	0	1519	11158	2022-04-28 5:46:59
6	5	24.01	76.48	249.92	98359	6	6	6	846	270	23	0	0	0	413	1	1523	12391	2022-04-28 5:47:27
7	6	24.06	76.34	249.92	98359	3	4	4	816	238	12	0	0	0	405	0	1520	13591	2022-04-28 5:47:54
8	7	24.1	76.18	249.92	98359	1	2	2	516	158	9	0	0	0	408	1	1520	14512	2022-04-28 5:48:21
9	8	24.14	76.09	249.92	98359	3	4	4	762	236	14	0	0	0	408	1	1515	15427	2022-04-28 5:48:48
10	9	24.18	75.8	249.92	98359	3	3	4	771	232	16	2	0	0	408	1	1523	16210	2022-04-28 5:49:15
11	10	24.2	75.72	249.92	98359	2	3	4	636	198	29	2	2	0	400	0	1521	16922	2022-04-28 5:49:42
12	11	24.23	75.5	250.09	98357	3	3	3	717	208	0	0	0	0	413	1	1521	17507	2022-04-28 5:50:09
13	12	24.25	75.45	250.09	98357	3	3	3	660	202	11	0	0	0	413	1	1520	18054	2022-04-28 5:50:37
14	13	24.26	75.42	250.17	98356	2	3	3	690	211	14	0	0	0	417	2	1520	18245	2022-04-28 5:51:06
15	14	24.27	75.41	250.26	98355	2	2	2	606	180	4	0	0	0	414	2	1521	18909	2022-04-28 5:51:33
16	15	24.32	75.24	250.35	98354	3	3	3	648	205	2	0	0	0	414	2	1527	19295	2022-04-28 5:52:01
17	16	24.37	75.13	250.43	98353	2	2	4	633	202	8	4	0	0	413	1	1532	19660	2022-04-28 5:52:28
18	17	24.4	75.03	250.52	98352	2	3	3	684	217	16	0	0	0	417	2	1530	19981	2022-04-28 5:52:55
19	18	24.43	74.91	250.52	98352	3	4	4	720	232	19	0	0	0	413	1	1531	20333	2022-04-28 5:53:22
20	19	24.44	74.77	250.52	98352	3	4	5	663	211	14	2	2	0	403	0	1535	20534	2022-04-28 5:53:49
21	20	24.44	74.7	250.6	98351	3	3	3	705	223	0	0	0	0	405	0	1533	21012	2022-04-28 5:54:16
22	21	24.44	74.7	250.52	98352	4	4	5	804	252	25	2	0	0	417	2	1523	21248	2022-04-28 5:54:44
23	22	24.47	74.69	250.52	98352	2	2	7	612	190	19	7	5	2	400	0	1526	21490	2022-04-28 5:55:11
24	23	24.5	74.68	250.43	98353	3	3	4	687	220	9	2	2	0	410	1	1521	21446	2022-04-28 5:55:38
25	24	24.51	74.8	250.26	98355	3	3	3	690	204	4	0	0	0	410	1	1515	21737	2022-04-28 5:56:05
26	25	24.52	74.84	250.09	98357	3	3	5	702	221	18	2	2	2	408	1	1523	21851	2022-04-28 5:56:32

Figure 4.11: A data sample from our stored data in a My-SQL data base

4.4.4 Prediction

4.4.4.1 Neural network

For the deep Learning model, we fit this model for up to 250 Epochs by using early stopping as a callback in order to store the best weights of the neural network and the Fig 4.12 displays the loss function chart.

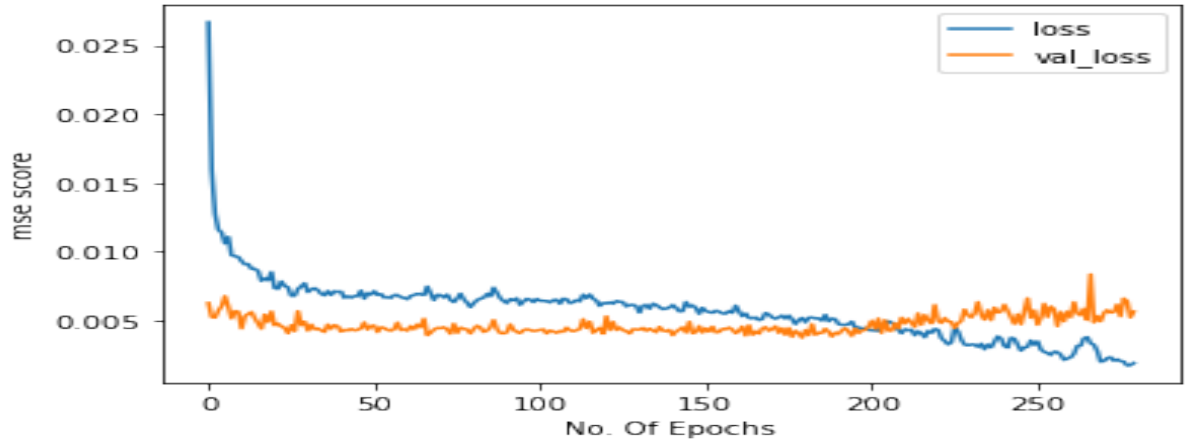


Figure 4.12: The loss function chart.

After that we will test the prediction by using test data and we denormalise the output and then plot it beside the true values, the Fig 4.13 shows the results.

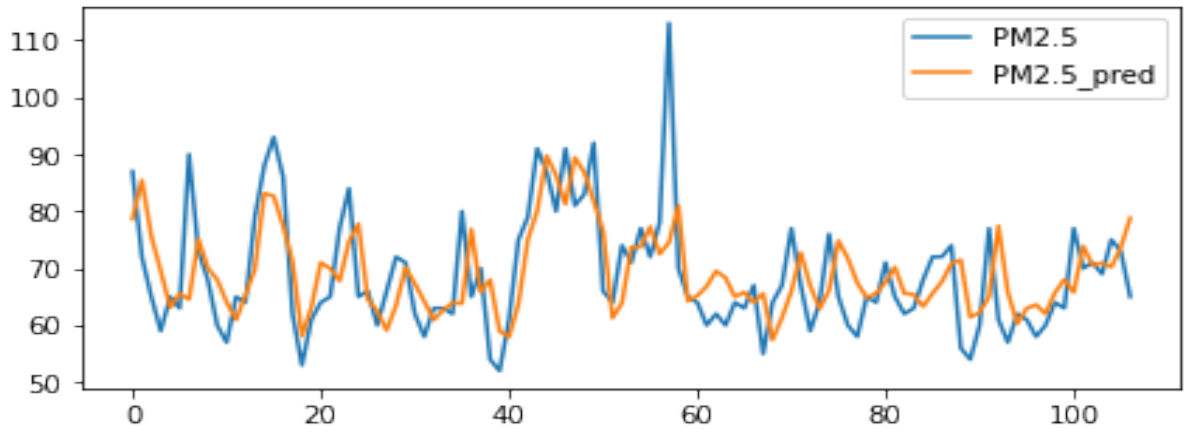


Figure 4.13: Forecast for more than 100 days.

As we can see, our neural network gave us an excellent result, we used two indicators, the mean absolute error (MAE) and the mean square error (RMSE). Our model has an RMSE of 8.56 and a MAE of 6.56.

4.4.4.2 SARIMA Model

after employing a rolling forecast technique, whenever we fit the model with the average of the current month and forecast the following month's mean, we plot the true value

in addition to the predicted one in the following figure 4.14.

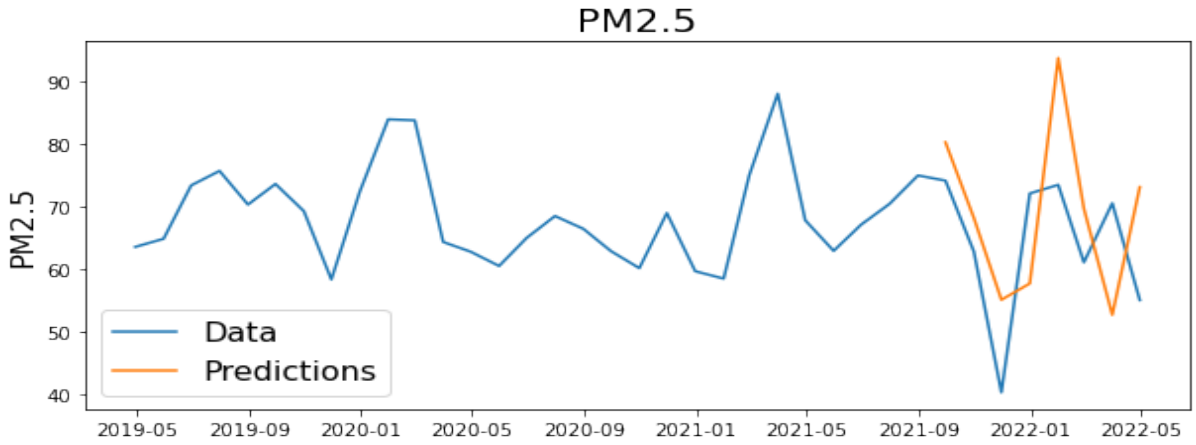


Figure 4.14: Forecasting result for 8 months.

As we can see, the results are not all that bad, they give us a good picture of the future values with the RMSE of 17.7 and the MAE of 13.18.

4.5 The prototype

The fig 4.15 shows the realised prototype with a future plans of improving it:



Figure 4.15: System's prototype1.0

4.6 Conclusion

In this chapter we presented the followed steps during the realisation of the prototype of this project , the software and hardware used with a brief of its use, the source code is consultable in our Github repository (SE cited April 2022).We also presented the edge device architecture describing its main working algorithm.

The functionalities, solutions and technologies that our system used can be summarised as follows:

-A hardware architecture that uses Raspberry Pi3 as a gateway (MQTT Broker), ESP32 as an edge device and Several sensors connected to the edge device providing us with data.

-A software architecture that we used, installed and configured in order to monitor, store and predict the data previously collected.

-Used an hybrid AI model in order to get the best result predicting data.

-An Mqtt protocol mainly used to assure communication between the different clients and the broker and which makes it easier for new clients to join in the network just by subscribing to topics.

-I2C, SPI and ADC protocols used in order to establish communication between the sensors and the edge device (ESP32).

This work enabled us to get a deep understanding idea on how the IoT systems work, communicate and collect data, also how the data should be treated before exposing it to a an AI model.

General Conclusion

General conclusion

During the achievement of this end of study project we had more knowledge about the neglected indoor pollution which can be caused from many indoor pollutants or even pollutants coming from the outside environment, the later considered as real danger for human's health especially the respiratory system, thing that drove us to consider monitor, predict the indoor air quality and collect data for future researches.

we were able to consume an important volume of understanding in the domain of Internet of Things (IoT) systems, which provides advanced services by interconnecting objects using existing or evolving inter-operable information, and communication technologies collecting data from the physical to the digital world and making it ready for any type of data analytic technologies.

We also learned better the domain of Artificial Intelligence (AI), which is a wide-ranging branch of computer science concerned with building smart machines capable of performing tasks that typically require human intelligence. We also approaches with one of its most known application which is prediction.

IoT these days is taking place among the most used technologies and the data collected are being unbelievably huge. The use of artificial intelligence gave it even more power and privilege.

We did a theoretical study about the IoT and its most used applications, its protocols and the challenges we can face. We did also a study about the Artificial Intelligence in the prediction field giving the most necessary notions in order to better understand how AI works.Which thereafter led us to chose the proper software and hardware tools to use.

In this end of study project we worked on an IoT system which is based on the MQTT using the raspberry pi3 as an MQTT broker, and the ESP32 (edge device) to which sensors are connected. Sensors provide data which will be consumed by MQTT clients in order to perform : monitoring, data storing and prediction.

A SARIMA forecast method has been used to provide medium-term predictions and to give us a view about how the data will be in the future, While the neural network has been used in order to get a close approximation of the value within the upcoming 24 hours.

For a future sight we are planning on adding more edge devices which means adding more sensors providing more informations about the indoor air quality.

Realising this project is having the wish to make the prototype industrialised and put it into the market.

Bibliographie

Albawi, Saad, Tareq Abed Mohammed, and Saad Al-Zawi (2017). "Understanding of a convolutional neural network." In: *2017 international conference on engineering and technology (ICET)*. Ieee, pp. 1–6.

Amihai, Ido et al. (2018). "An industrial case study using vibration data and machine learning to predict asset health." In: *2018 IEEE 20th Conference on Business Informatics (CBI)*. Vol. 1. IEEE, pp. 178–185.

Blomquist, Göran (1994). "Sampling of biological particles." In: *Analyst* 119.1, pp. 53–56.

Box, George EP et al. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.

Brandimarte, Paolo (2013). *Numerical methods in finance and economics: a MATLAB-based introduction*. John Wiley & Sons.

Brownlee, Jason (2018). *Deep learning for time series forecasting: predict the future with MLPs, CNNs and LSTMs in Python*. Machine Learning Mastery.

Cryer, Jonathan D and Kung-Sik Chan (2008). *Time series analysis: with applications in R*. Vol. 2. Springer.

Dorffner, Georg (1996). "Neural networks for time series processing." In: *Neural network world*. Citeseer.

Elman, Jeffrey L (1990). "Finding structure in time." In: *Cognitive science* 14.2, pp. 179–211.

Farhan, Laith et al. (2017). "A survey on the challenges and opportunities of the Internet of Things (IoT)." In: *2017 Eleventh International Conference on Sensing Technology (ICST)*, pp. 1–5.

Farhan, Laith et al. (2018a). "A Concise Review on Internet of Things (IoT) -Problems, Challenges and Opportunities." In: *2018 11th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP)*, pp. 1–6.

— (2018b). "A Concise Review on Internet of Things (IoT) -Problems, Challenges and Opportunities." In: *2018 11th International Symposium on Communication Systems, Networks Digital Signal Processing (CSNDSP)*, pp. 1–6.

Farhan, Laith et al. (2018c). "Towards green computing for Internet of things: Energy oriented path and message scheduling approach." In: *Sustainable Cities and Society* 38, pp. 195–204.

Gers, Felix A, Jürgen Schmidhuber, and Fred Cummins (2000). "Learning to forget: Continual prediction with LSTM." In: *Neural computation* 12.10, pp. 2451–2471.

Gourieroux, Christian, Alain Monfort, and Gourieroux Christian (1997). *Time series and dynamic models*. Vol. 3. Cambridge University Press.

Gubbi, Jayavardhana et al. (2013). "Internet of Things (IoT): A vision, architectural elements, and future directions." In: *ArXiv* abs/1207.0203.

Hochreiter, Sepp and Jürgen Schmidhuber (1997). "Long short-term memory." In: *Neural computation* 9.8, pp. 1735–1780.

Hyndman, Rob J and George Athanasopoulos (2018). *Forecasting: principles and practice*. OTexts.

Karlik, Bekir and A Vehbi Olgac (2011). "Performance analysis of various activation functions in generalized MLP architectures of neural networks." In: *International Journal of Artificial Intelligence and Expert Systems* 1.4, pp. 111–122.

Kelleher, John D (2019). *Deep learning*. MIT press.

Khan, Asifullah et al. (2020). "A survey of the recent architectures of deep convolutional neural networks." In: *Artificial intelligence review* 53.8, pp. 5455–5516.

Ko, Jeonggil et al. (2011). "Connecting low-power and lossy networks to the internet." In: *IEEE Communications Magazine* 49.4, pp. 96–101.

Kulkarni, Santosh and Sanjeev Kulkarni (2017). "Communication models in internet of things: a survey." In: *International Journal of Science Technology & Engineering* 3.11, pp. 87–91.

Ma, Xin and Wei Luo (2008). "The analysis of 6LoWPAN technology." In: *2008 IEEE Pacific-Asia Workshop on Computational Intelligence and Industrial Application*. Vol. 1. IEEE, pp. 963–966.

Maroni, Marco, Bernd Seifert, and Thomas Lindvall (1995). *Indoor air quality: a comprehensive reference book*. Elsevier.

O'Shea, Keiron and Ryan Nash (2015). "An introduction to convolutional neural networks." In: *arXiv preprint arXiv:1511.08458*.

Palattella, Maria Rita et al. (2013). "Standardized Protocol Stack for the Internet of (Important) Things." In: *IEEE Communications Surveys Tutorials* 15.3, pp. 1389–1406.

Patel, Keyur K. and Sunil M Patel (2016). "Internet of Things-IOT : Definition , Characteristics , Architecture , Enabling Technologies , Application & Future Challenges." In.

Rose, Karen, Scott Eldridge, and Lyman Chapin (2015). "The internet of things: An overview." In: *The internet society (ISOC)* 80, pp. 1–50.

Sobin, CC (2020). "A survey on architecture, protocols and challenges in IoT." In: *Wireless Personal Communications* 112.3, pp. 1383–1429.

Sønderby, Søren Kaae and Ole Winther (2014). "Protein secondary structure prediction with long short term memory networks." In: *arXiv preprint arXiv:1412.7828*.

Srivastava, Nitish et al. (2014). "Dropout: a simple way to prevent neural networks from overfitting." In: *The journal of machine learning research* 15.1, pp. 1929–1958.

Staudemeyer, Ralf C and Eric Rothstein Morris (2019). "Understanding LSTM–a tutorial into long short-term memory recurrent neural networks." In: *arXiv preprint arXiv:1909.09586*.

Xu, Guixian et al. (2019). "Sentiment analysis of comment texts based on BiLSTM." In: *Ieee Access* 7, pp. 51522–51532.

Yu, Yong et al. (2019). "A review of recurrent neural networks: LSTM cells and network architectures." In: *Neural computation* 31.7, pp. 1235–1270.

Zheng, Haifeng et al. (2020). “A hybrid deep learning model with attention-based conv-LSTM networks for short-term traffic flow prediction.” In: *IEEE Transactions on Intelligent Transportation Systems* 22.11, pp. 6910–6920.

Webographie

aqicn (n.d.). *PM2.5 DATA*.

Connectwave (n.d.). *Les réseaux IoT*. URL: https://www.connectwave.fr/techno-appli-iot/iot/les-reseaux-iot/.

raspberry-pi-os (cited June 2022). *Raspberry-pi-os*. URL: https://www.raspberrypi.com/software/.

SE (cited April 2022). *Github*. URL: https://github.com/NadirMohamedCHERFIA/IoT_Air_Quality.git.

Sparkplug (cited June 2022). *Sparkplug Specifications*. URL: https://www.eclipse.org/tahu/spec/Sparkplug%5C%20Topic%5C%20Namespace%5C%20and%5C%20State%5C%20ManagementV2.2-with%5C%20appendix%5C%20B%5C%20format%5C%20-%5C%20Eclipse.pdf.

My-SQL-workbench (cited June 2022). *My-SQL-workbench*. URL: https://fr.wikipedia.org/wiki/MySQL_Workbench.

Appendices

Appendix A

Définitions

A.1 Big data challenges (5V)

A.1.0.1 Velocity

Refers to the speed with which data is collected, transferred and processed. The speed of data processing is different depending on the type of application. For some applications, incoming data can be processed in a short period of time, while other applications require real-time processing, such as analytical programs.(Farhan et al. 2018a)

A.1.0.2 Variety

Refers to the different types of data collected by end devices such as smartphones, machines, sensors, etc. The content of the data is unstructured, of different types, such as audio, video, images, XML format plain text, CSV format, etc. The variety of data must be organized and processed in a meaningful and consistent way.(Farhan et al. 2018a)

A.1.0.3 Varacity

Means ensuring that the data collected and stored is accurate. This may involve filtering out any unwanted or corrupted data in order to improve the quality of the application.(Farhan et al. 2018a)

A.1.0.4 Volume

Is the amount of all types of data that are collected, stored, retrieved, updated from different sources. The IoT generates a huge amount of data that is growing exponentially. The question is whether we can integrate volume and velocity together.(Farhan et al. 2018a)

A.1.0.5 Value

Once the big data is accurately collected, the next step is to extract value from the data. Therefore, various algorithms such as feature extraction, trend analysis using AI to make informed decisions, within the required time frame, is another challenge.(Farhan et al. 2018a)

A.2 Protocols

A.2.1 IPv6 and the Internet of Things

Though they differ about the exact numbers, most technology observers agree that billions of additional devices – from industrial sensors to home appliances and vehicles – will be connected to the Internet between now and 2025. As the Internet of Things continues to grow, devices that require true end-to-end Internet connectivity will not be able to rely on IPv4, the protocol most Internet services use today. They will need a new enabling technology: IPv6.

IPv6 is a long-anticipated upgrade to the Internet's original fundamental protocol – the Internet Protocol (IP), which supports all communications on the Internet. IPv6 is necessary because the Internet is running out of original IPv4 addresses. While IPv4 can support 4.3 billion devices connected to the Internet, IPv6 with 2 to the 128th power addresses, is for all practical purposes inexhaustible. This represents about 340 trillion, trillion, trillion addresses, which more than satisfies the demand of the estimated 100 billion IoT devices going into service in the coming decades.

Given the anticipated longevity of some of the sensors and other devices imagined for the Internet of Things, design decisions will affect the utility of solutions decades from now. Key challenges for IoT developers are that IPv6 is not natively interoperable with IPv4 and most low-cost software that is readily available for embedding in IoT devices implements only IPv4. Many experts believe, however, that IPv6 is the best connectivity option and will allow IoT to reach its potential.

A.3 Softwares

A.3.1 My-SQL workbench

MySQL Workbench (formerly MySQL administrator) is a MySQL database management and administration software created in 2004. (My-SQL-workbench cited June 2022)

A.3.2 Raspberry pi imager

Raspberry Pi Imager is the quick and easy way to install Raspberry Pi OS and other operating systems to a microSD card, ready to use with your Raspberry Pi.(raspberry-pi-os cited June 2022)

Appendix B

Tables

B.1 Major indoor pollutants and emission sources

Pollutant	Major emission sources
Allergens	House dust, domestic animals, insects
Asbestos	Fire retardant materials, insulation
Carbon dioxide	Metabolic activity, combustion activities, motor vehicles in garages
Carbon monoxide	Fuel burning, boilers, stoves, gas or kerosene heaters, tobacco smoke
Formaldehyde	Particleboard, insulation, furnishings
Micro-organisms	People, animals, plants, air conditioning systems
Nitrogen dioxide	Outdoor air, fuel burning, motor vehicles in garages
Organic substances	Adhesives, solvents, building materials, volatilisation, combustion, paints, tobacco smoke
Ozone	Photochemical reactions
Particles	Re-suspension, tobacco smoke, combustion products
Polycyclic aromatic hydrocarbons	Fuel combustion, tobacco smoke
Pollens	Outdoor air, trees, grass, weeds, plants
Radon	Soil, building construction materials (concrete, stone)
Fungal spores	Soil, plants, foodstuffs, internal surfaces
Sulphur dioxide	Outdoor air, fuel combustion

Table B.1: Major indoor pollutants and emission sources(Farhan et al. 2018b)

B.2 Outdoor sources of major indoor air pollutants

Pollutant	Percentage of emissions associated with industry	Percentage of emissions associated with transport
Benzene	32	65
Carbon monoxide (CO)	3	90
Oxides of nitrogen (NOx)	38	49
Particulates (PM10)	56	25
Sulphur dioxide (SO2)	90	2
Volatile organic compounds (VOCs)	52	34

Table B.2: Outdoor sources of major indoor air pollutants(Farhan et al. 2018b)

Appendix C

Figures

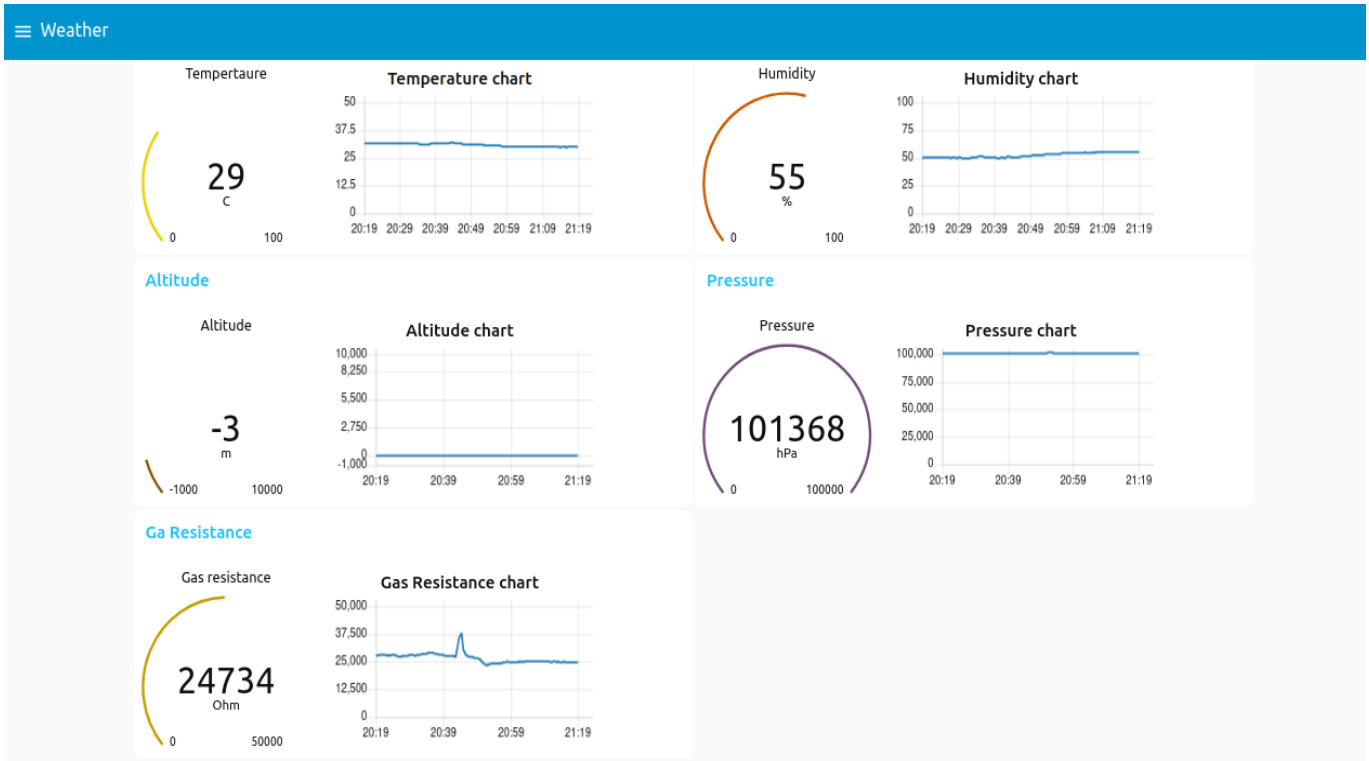


Figure C.1: TAB 1 NODE RED dashboard monitoring

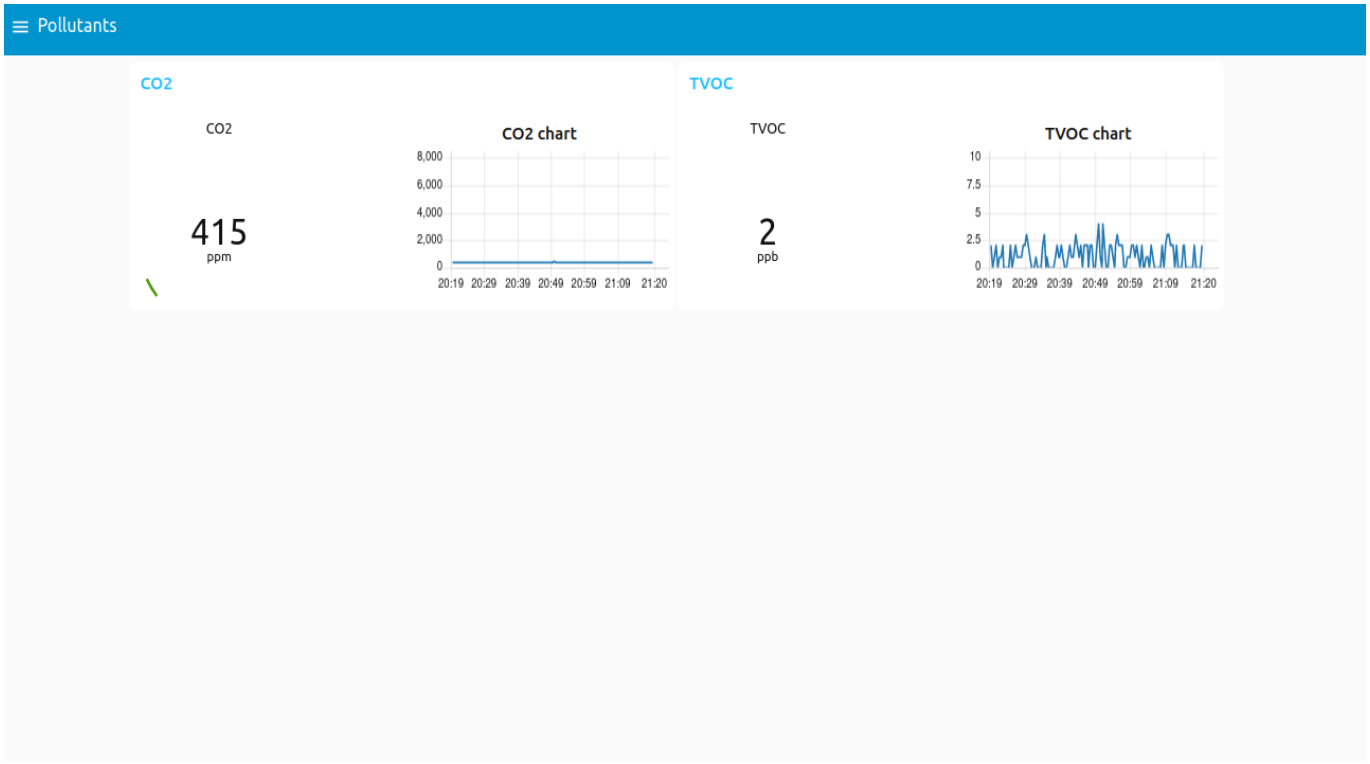


Figure C.2: TAB 2 NODE RED dashboard monitoring

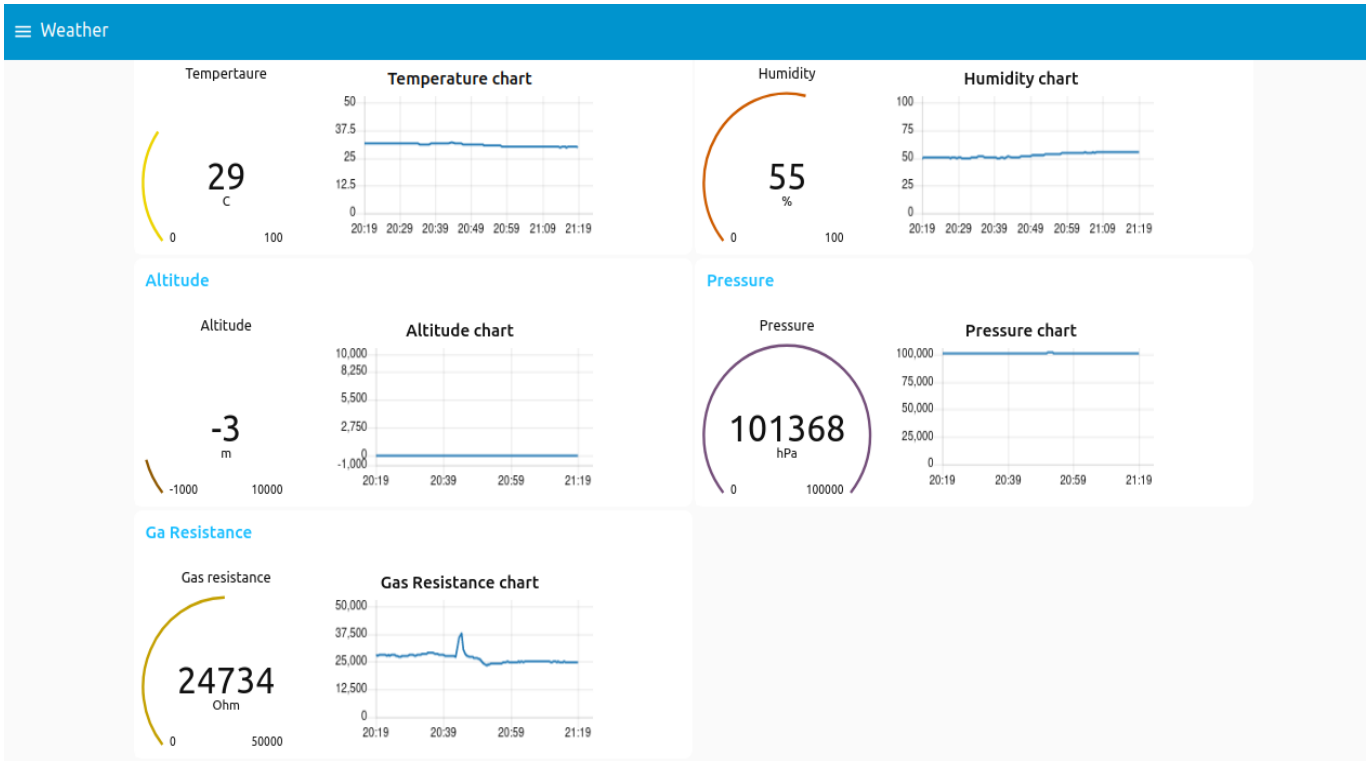


Figure C.3: TAB 3 NODE RED dashboard monitoring

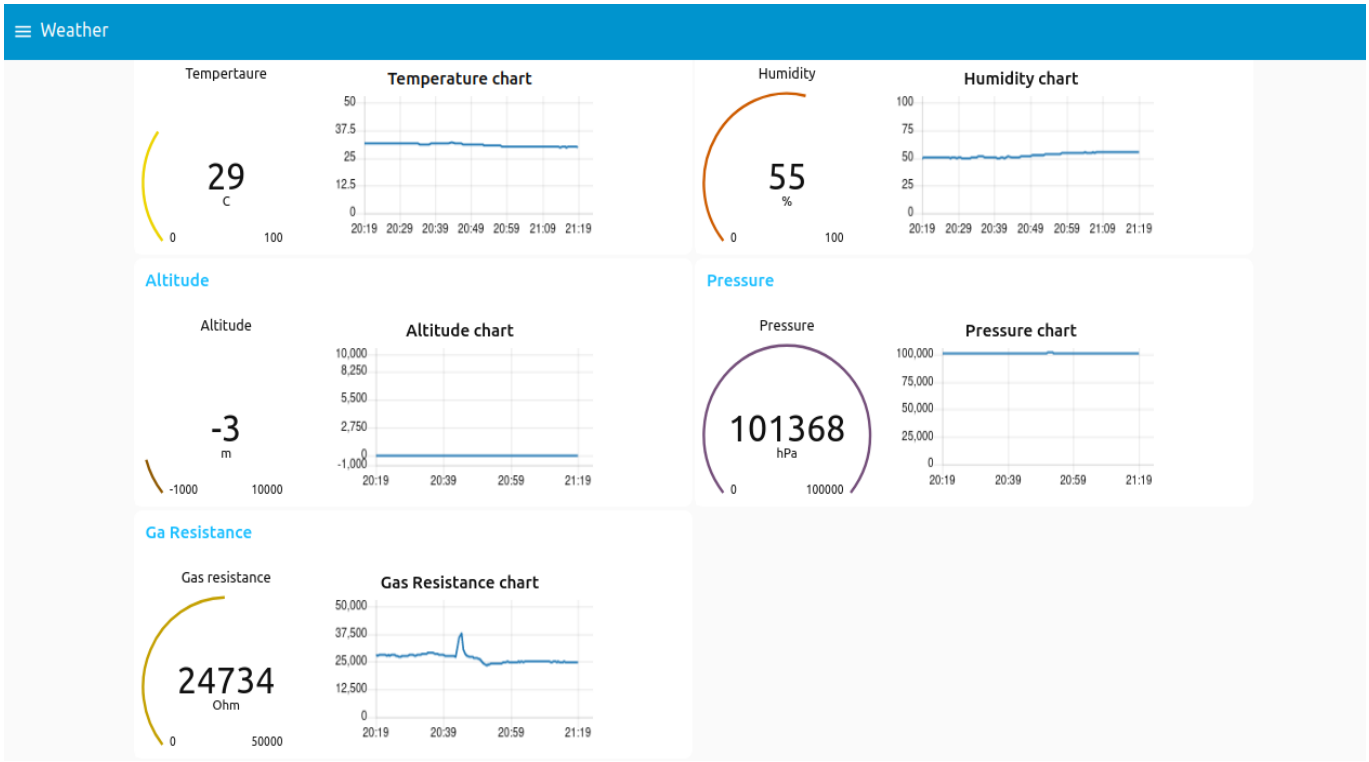


Figure C.4: TAB 4 NODE RED dashboard monitoring

C.1 Vs Code interface

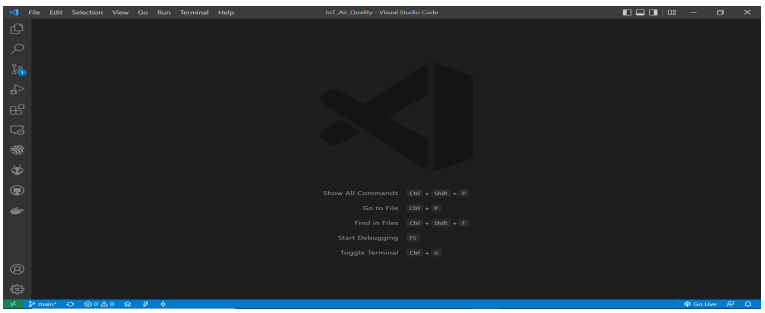


Figure C.5: Visual Studio Code Interface

C.2 Platform io Interface in VS Code

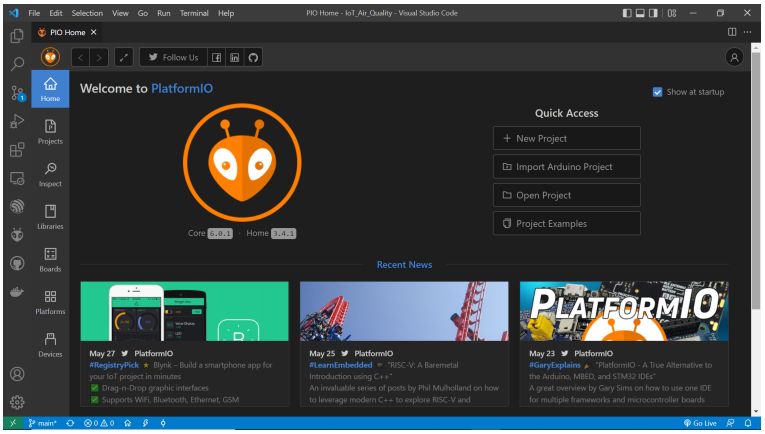


Figure C.6: Platform io Interface in VS Code

C.3 IA Architecure

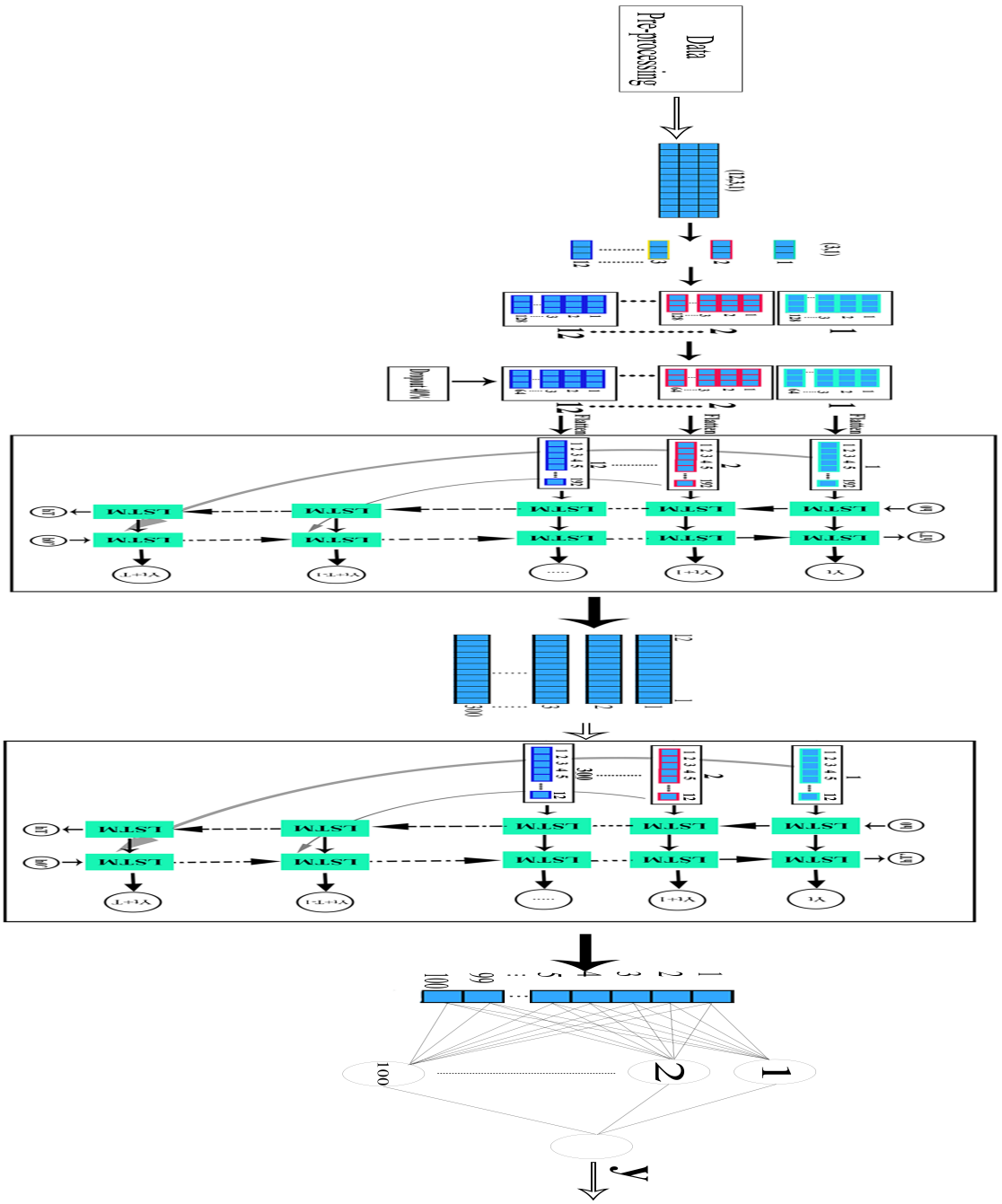


Figure C.7: Network structure of the proposed hybrid CNN-LSTM model

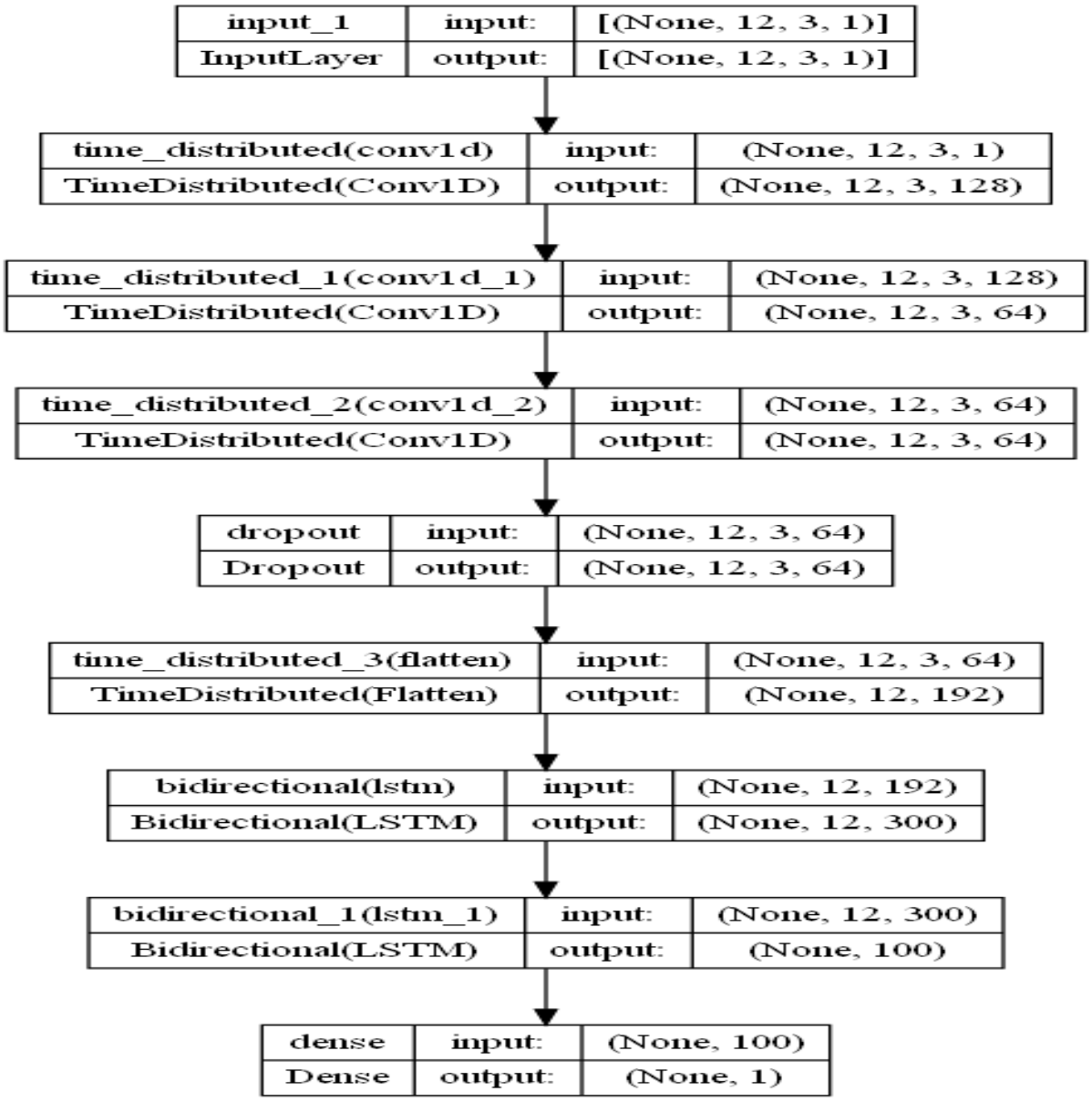


Figure C.8: Internal network structure of the proposed hybrid CNN-LSTM model generated by anaconda platform.

Appendix D

Features and specs

D.1 Raspberry Pi 3's specs

The following are the Raspberry Pi 3's specs:.[1]. -Chipset: Broadcom BCM2837

-CPU: 1.2GHz quad-core 64-bit ARM cortex A53.

-Ethernet : 10/100 (Max throughput 100Mbps).

-USB: Four USB 2.0 with 480Mbps data transfer.

-Storage: MicroSD card or via USB-attached storage.

-Wireless: 802.11n Wireless LAN (Peak transmit/receive.

-throughput of 150Mbps), Bluetooth 4.1.

-Graphics: 400MHz VideoCore IV multimedia.

-Memory: 1GB LPDDR2-900 SDRAM.

-Expandability: 40 general purpose input-output pins.

-Video: Full HDMI port.

-Audio: Combined 3.5mm audio out jack and composite video.

-Camera interface (CSI).

-Display interface (DSI).

D.2 ESP32 features

Processors

-CPU: Xtensa dual-core (or single-core) 32-bit LX6 microprocessor, operating at 160 or 240 MHz and performing at up to 600 DMIPS.

-Ultra low power (ULP) co-processor.

[1] https://www.zdnet.com/article/what-is-the-raspberry-pi-3-everything-you-need-to-know-about-the- 2022.

Memory

-320 KiB RAM, 448 KiB ROM.

Wireless connectivity -Wi-Fi: 802.11 b/g/n.

-Bluetooth: v4.2 BR/EDR and BLE (shares the radio with Wi-Fi).

Peripheral interfaces

-34 × programmable GPIOs.

-12-bit SAR ADC up to 18 channels.

-2 × 8-bit DACs.

-10 × touch sensors (capacitive sensing GPIOs).

-4 × SPI.

-2 × I²S interfaces.

-2 × I²C interfaces.

-3 × UART. -SD/SDIO/CE-ATA/MMC/eMMC host controller.

-SDIO/SPI slave controller Ethernet MAC interface with dedicated DMA and planned IEEE 1588 Precision Time Protocol support[2].

-CAN bus 2.0.

-Infrared remote controller (TX/RX, up to 8 channels).

-Motor PWM.

-LED PWM (up to 16 channels).

-Hall effect sensor.

-Ultra low power analog pre-amplifier.

Security

-IEEE 802.11 standard security features all supported, including WPA, WPA2, WPA3 (depending on version)[3]. and WLAN Authentication and Privacy Infrastructure (WAPI).

-Secure boot.

-Flash encryption.

-1024-bit OTP, up to 768-bit for customers.

-Cryptographic hardware acceleration: AES, SHA-2, RSA, elliptic curve cryptography (ECC), random number generator (RNG).

Power management

-Internal low-dropout regulator.

-Individual power domain for RTC.

-5 A deep sleep current.

-Wake up from GPIO interrupt, timer, ADC measurements, capacitive touch sensor

[2] https://github.com/espressif/esp-idf/issues/1223#ref-issue-482153951,Mai 2022.
[3] https://www.espressif.com/en/support/documents/certificates,Mai 2022.

interrupt.

D.3 PMSA00I sensor module specifications:

-Particle Range of measurement: $0.3 \rightsquigarrow 1.0, 1.0 \rightsquigarrow 2.5, 2.5 \rightsquigarrow 10 Micrometer$.

-Particle Counting Efficiency: 50% @ 0.3 m 98% @ >=0.5 m.

-Particle Effective Range (PM2.5 standard): $0 \rightsquigarrow 500 g/m^3$.

-Particle Maximum Range (PM2.5 standard): $\geq 1000 g/m^3$.

-Particle Resolution: 1 g/m^3.

-Particle Standard Volume 0.1 Liter.

-Single Response Time 1 second.

-Total Response Time $\leq 10 seconds$.

-DC Power Supply Typ: 5.0V Min: 4.5V Max: 5.5V.

-Active Current$\leq 100 mA$.

-Standby Current$\leq 200 A$.

-Interface Level: 3.3V logic, L <0.8V, H >2.7V.

-Working Temperature Range -10 $\rightsquigarrow$ 60℃..

-Working Humidity Range $0 \rightsquigarrow$ 99%.

-Storage Temperature Range -40 80 ℃..

-MTTF $\geq 3 Year$.

-I2C address 0x12 (cannot be changed).

-Product Dimensions: 51.0mm x 35.5mm x 13.6mm / 2.0” x 1.4” x 0.5”.

-Product Weight: 28.0g / 1.0oz.

D.4 SEN0335 sensor Specification

-Operating Voltage: 3.3V$\rightsquigarrow$ 5.5V.

-Working Current: <20mA.

CCS811 Parameter

-Preheat Time: <15s.

-I2C Address: 0x5A(in default)/0X5B.

-Operating Temperature Range: -40℃ $\rightsquigarrow$ 5℃.

-Operating Humidity Range: 10%RH$\rightsquigarrow$ 95%RH.

-eCO2 Measuring Range: 400ppm$\rightsquigarrow$ 8000ppm.

-TVOC Measuring Range: 0ppb$\rightsquigarrow$ 1100ppb.

BME280 Parameter

-I2CAddress 0x76(in default)/0X77.

-Operating Temperature: -40℃⇝ 85℃.

-Temperature Measuring Range: -40℃ ⇝ 85℃.

-resolution of 0.1℃, deviation of ±0.5℃.

-Humidity Measuring Range: 0⇝ 100%RH.

-resolution of 0.1%RH, deviation of ±2%RH.

-Pressure Measuring Range: 300⇝ 1100hPa.

D.5 BME680 specifications

Package dimensions

-8-Pin LGA with metal.

-3.0 x 3.0 x 0.93 mm^3.

Operation range (full accuracy)

-Pressure: 300...1100 hPa.

-Humidity 0…100%.

-Temperature: -40…85°C.

Supply voltage VDDIO

- 1.2 ... 3.6 V.

Supply voltage VDD

- 1.71 ... 3.6 V.

Interface

-I2C and SPI.

D.6 MQ135 Specfications

-High Sensitivity.

-High sensitivity to Ammonia, Sulfide and Benze.

-Stable and Long Life.

-Detection Range: 10 - 300 ppm NH3, 10 - 1000 ppm Benzene, 10 - 300 Alcohol.

-Heater Voltage: 5.0V.

-Dimensions: 18mm Diameter, 17mm,High excluding pins, Pins - 6mm High.

-Long life and low cost.

Abstract

The Internet of Things (IoT) is widely used in several applications which mainly revolves around designing and creating platforms capable of collecting data, storing it and synthesizing it from where its implemented.And like any other technologies its has its own challenges and impediments like connectivity challenges, big data, energy consumption ..etc.

Our work presented in this dissertation consists of developing an IoT platform capable of collecting, monitoring, sroring and predicting Indoor air quality using sensors connected to an edge device (ESP32) acting as an MQTT client which obviously uses MQTT protocol to send data to the gateway which is a Raspberry Pi3 having the role of an MQTT broker, and having also other clients subscribed to topics on the broker which will allow the monitoring, storing and also for predicting the next day's data and the mean of the whole next month using Artificial Intelligence models.

Keywords : Internet of Things, Artificial Intelligence, Gateway, Air quality, MQTT broker, MQTT clients, Edge device, Prediction.

Résumé

L'Internet des objets (IoT) est largement utilisé dans plusieurs applications qui tournent principalement autour de la conception, de la création de plates-formes capables de collecter des données, de les stocker et de les synthétiser à partir de l'endroit où elles sont mises en œuvre.

Le travail présenté dans ce péesent mémoire consiste à développer une plateforme IoT capable de collecter, surveiller, mesurer et prédire la qualité de l'air intérieur en utilisant des capteurs connectés à edge device (ESP32) agissant comme un client MQTT utilisant le protocole MQTT pour envoyer des données au passerelle Raspberry Pi3 ayant le rôle d'un broker MQTT et ayant également d'autres clients souscrits à des sujets sur le broker qui permettront le suivi, le stockage et aussi la prédiction des données du jour suivant et de la moyenne du mois suivant utilisant des modéles de l'intelligence artificielle.

Mots clés : Internet des Objets, Intelligence Artificialle, passerelle, Qualité de l'air, MQTT broker, clients MQTT, Edge device, Prédiction.

ملخص

يتكون عملنا المقدم في هذه الأطروحة من تطوير منصة إنترنت الأشياء قادرة على جمع ومراقبة بيانات جودة الهواء الداخلي والتنبؤ به باستخدام أجهزة استشعار متصلة إلى device edge (ESP32) يعمل كعميل MQTT والذي يستخدم بروتوكول MQTT لإرسال البيانات إلى البوابة وهي Pi3 Raspberry التي تلعب دور broker MQTT ، وكذلك وجود عملاء آخرين مشتركين في topics عن طريق الوسيط مما سيسمح ذلك بالمراقبة والتخزين وأيضًا للتنبؤ ببيانات اليوم التالي ومتوسط كامل الشهر المقبل باستخدام نماذج الذكاء الاصطناعي.

كلمات مفتاحية : انترنت الأشياء,الذكاء الاصطناعي, البوابة, جودة الهواء, التنبؤ

عملاء MQTT

MQTT Broker

Edge Device