PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

MINISTRY OF HIGHER EDUCATION AND RESEARCH SCIENTIST

Faculty of Technology

Department of Electrical Engineering

Dissertation

ACADEMIC MASTER

Domain: Science and Technology

Division: Telecommunications

Specialty: Telecommunication Systems

Presented by:

- ***MAHMOUDI Tahar Mortadha***
- ***SOBTI Ala Eddine***

Entitled:

Implementation of an Intelligent System for Air Quality Monitoring

Dissertation Submitted in Partial Fulfillment of the Requirements for the Master Degree

Publicly defended in: 03/06 /2024

Board of Examiners:

Pr. AJGOU Riadh	**Chairman**
Dr. HIMA Abdelkader	**Supervisor**
Dr. AOURAGH Nabil	**Examiner**

***Academic year*: 2023/2024**

بسم الله الرحمن الرحيم

Dedication

To our beloved families,

To our parents and siblings,

To our friend's,

Your unwavering support, constant encouragement, and endless inspiration have been our guiding lights. This work is a testament to the love and strength you have given us throughout this journey.

With all our gratitude and love.

Acknowledgements

First and foremost, we would like to express our deepest gratitude to Allah, the Almighty, whose blessings have enabled us to complete this work.

Our sincere thanks go to our supervisor, Mr. HIMA Abdelkader, for his consistent support, valuable guidance, and constructive feedback. His expertise and encouragement have been instrumental in the successful completion of this work.

We are also immensely grateful to the members of our jury. Our heartfelt appreciation goes to them for their willingness to examine our thesis and provide insightful comments.

We take this opportunity to extend our gratitude to all our teachers who have contributed to our education with their collaboration, availability, and kindness.

We are deeply thankful to our families for their unwavering support and encouragement. We can never fully express our gratitude to our parents, who have always gone above and beyond to ensure our success and well-being.

Finally, we would like to thank everyone who has helped us directly or indirectly throughout this journey, especially our colleagues from the Electrical Engineering Department. Your support and camaraderie have been invaluable.

ملخص:

يتناول هذا العمل تصميم وتنفيذ نظام ذكي لرصد جودة الهواء باستخدام أردوينو ومجموعة من أجهزة الاستشعار. يتكون النظام من جانب العتاد الصلب الذي يشمل أردوينو نانو، ومصدر تغذية القدرة، ولوحة الربط، وأسلاك التوصيل، وأجهزة استشعار الغازات (MQ-7، MQ-9، MQ-135)، ومستشعر درجة الحرارة والرطوبة (DHT11)، ووحدة ESP01 لاتصال Wi-Fi، ووحدة بطاقة SD الصغيرة لتخزين البيانات. أما الجانب البرمجي فيتضمن برنامج أردوينو IDE 2، ومنصة ThingSpeak لجمع وتحليل البيانات، ونماذج الذكاء الاصطناعي في Google Colab للتنبؤ بجودة الهواء. يوفر النظام بيانات حول تركيزات الغازات المختلفة ودرجات الحرارة والرطوبة، ويتم إرسال هذه البيانات عبر Wi-Fi إلى ThingSpeak لعرضها وتحليلها. بعد ذلك، تُستخدم نماذج الذكاء الاصطناعي مثل الشبكات العصبية الاصطناعية وغيرها لتحليل البيانات والتنبؤ بمستويات جودة الهواء المستقبلية. يتم تقييم أداء النماذج المختلفة باستخدام مقاييس مثل متوسط الخطأ المطلق والخطأ النسبي المطلق والجذر التربيعي لمتوسط مربعات الخطأ لتحديد أفضل نموذج للتنبؤ بجودة الهواء.

الكلمات المفتاحية: أردوينو، رصد جودة الهواء، أجهزة استشعار الغازات، مستشعر درجة الحرارة والرطوبة، ThingSpeak، الذكاء الاصطناعي، التنبؤ، الشبكات العصبية الاصطناعية.

Abstract:

This work focuses on the design and implementation of an intelligent air quality monitoring system using Arduino and a set of sensors. The system consists of a hardware aspect that includes an Arduino Nano, a power supply, a breadboard, jumper wires, gas sensors (MQ-7, MQ-9, MQ-135), a temperature and humidity sensor (DHT11), an ESP01 unit for Wi-Fi connectivity, and a Micro SD card module for data storage. The software aspect involves the Arduino IDE 2 software, the ThingSpeak platform for data aggregation and analysis, and AI models in Google Colab for air quality prediction. The system provides data on concentrations of various gases, temperature, and humidity, which are transmitted over Wi-Fi to ThingSpeak for visualization and analysis. Subsequently, AI models such as Artificial Neural Networks and others are employed to analyze the data and predict future air quality levels. The performance of different models is evaluated using metrics like Mean Absolute Error, Mean Absolute Percentage Error, and Root Mean Squared Error to determine the best model for air quality prediction.

Keywords: Arduino, air quality monitoring, gas sensors, temperature and humidity sensor, ThingSpeak, Artificial Intelligence, prediction, Artificial Neural Networks.

Table of contents

Dedication ____ *I*

Acknowledgements ____ *II*

ملخص ____ *III*

Abstract ____ *III*

Table of contents ____ *IV*

List of Figures ____ *VI*

list of Tables ____ *VI*

Abbreviation's list ____ *VII*

General Introduction ____ *VIII*

Chapter I : Introduction to Arduino Programming and Development Environment

I.1 Introduction ____ *2*

I.2 What is Arduino ? ____ *2*

I.2.1 Types of Arduino Boards – Comparison on Specification and Features ____ *3*

I.2.2 Anatomy of an Arduino Board ____ *5*

I.3 Programming in Arduino IDE 2 Software ____ *6*

I.3.1 Installing Arduino IDE 2 ____ *6*

I.3.2 Arduino is programmed with C and C++ ____ *7*

I.3.3 Structure ____ *7*

I.4 Conclusion ____ *8*

Chapter II : Artificial Intelligence & the Internet of Things

II.1 Introduction ____ *11*

II.2 Artificial intelligence -AI- ____ *11*

II.2.1 What is AI ? ____ *11*

II.2.2 Why is AI Crucial ? ____ *11*

II.2.3 How Does AI Work ? ____ *12*

II.2.4 AI Techniques ____ *13*

II.3 Internet of Things ____ *16*

II.3.1 What Is IoT ? ____ *16*

II.3.2 Applications of IoT Systems ____ *17*

II.3.3 Security and Privacy ____ *18*

II.4 Conclusion ____ *19*

Chapter III : System design and implementation

III.1 Introduction 21

III.2 Basic components of the system 21

III.2.1 Hardware aspect 22

III.2.2 Software aspect 28

III.3 Step-by-Step Guide 30

III.4 How the system works ? 37

III.5 The final desired model 38

III.6 Conclusion 39

Chapter IV : Discussion of Results

IV.1 Introduction 41

IV.2 Descriptive Statistics for Environmental Measurements 41

IV.3 Comparative Analysis of Predictive Models 43

IV.4 Conclusion 51

General Conclusion 52

Bibliographies 54

List of Figures

Figure I.1: Types of Arduino Boards 3

Figure I.2: Key components of an Arduino board 5

Figure I.3: Interface Arduino IDE 2 software 6

Figure II.1: A conceptual view of artificial intelligence 11

Figure III.1: Arduino Nano 22

Figure III.2: MQ-7 for carbon monoxide (CO) 22

Figure III.3: MQ-9 Carbon Monoxide, Methane and LPG Gas Sensor 23

Figure III.4: MQ 135 for Air Quality 23

Figure III.5: MQ Gas Sensor Module Pinout 24

Figure III.6: DHT11 Sensor 24

Figure III.7: ESP-01 Wi-Fi Module 25

Figure III.8: Micro SD Card Adapter Module 26

Figure III.9: Breadboard power supply module with cable USB (Male-Male) 26

Figure III.10: Breadboard 27

Figure III.10: Jumper wires 28

Figure III.10: ThingSpeak IoT logo 28

Figure III.11: Colab AI logo 29

Figure III.11: The first model of the system 31

Figure III.12: Schematic overview of the smart system to study air quality 37

Figure III.12: The final desired model for an intelligent air quality monitoring system 38

Figure IV.1: ANN Predictions and Actual CO Concentrations 44

Figure IV.2: K-NN Predictions and Actual CO Concentrations 44

Figure IV.3: CNN Predictions and Actual CO Concentrations 45

Figure IV.4: CNN-LTSM Predictions and Actual CO Concentrations 45

Figure IV.5: CNN-LTSM-BOA Predictions and Actual CO Concentrations 46

Figure IV.6: LTSM Predictions and Actual CO Concentrations 46

Figure IV.7: Random Forest Predictions and Actual CO Concentrations 47

Figure IV.8: Multiple Linear Regression Predictions and Actual CO Concentrations 47

Figure IV.9: Support Vector Machine (SVM) Predictions and Actual CO Concentrations 48

List of Tables

Table I.1: Comparison of Arduino Board Types, Specifications and Features 4
Table III.1: Wiring/connection table for various components 31
Table III.2: Model Descriptions and Capabilities 36
Table IV.1: Descriptive Statistics for Environmental Measurements: CO, CH4, NH3, Temperature, and Humidity 41
Table IV.2: Performance evaluation of different models 48

Abbreviation's list

AI - Artificial Intelligence

ANN - Artificial Neural Network

BOA - Bayesian Optimization Algorithm

CH4 - Methane

CNN - Convolutional Neural Network

CO - Carbon Monoxide

IDE - Integrated Development Environment

IoT - Internet of Things

k-NN - k-Nearest Neighbors

LSTM - Long Short-Term Memory

MAE - Mean Absolute Error

MAPE - Mean Absolute Percentage Error

MSLE - Mean Squared Logarithmic Error

NH3 - Ammonia

RH - Relative Humidity

RMSE - Root Mean Squared Error

SVM - Support Vector Machine

General Introduction

Air pollution stands as a formidable challenge affecting both public health and the environment on a global scale. The imperative need for precise monitoring and analysis of air quality parameters is evident for formulating effective strategies to mitigate its impact. This thesis endeavors to address this pressing need through the development, implementation, and evaluation of an intelligent air quality monitoring system. Employing a fusion of the Internet of Things (IoT), artificial intelligence (AI), and advanced data analytics techniques, this system represents a significant advancement in environmental monitoring technology.

The central issue at hand revolves around the shortcomings of existing methods in furnishing the requisite granularity and predictive capabilities essential for proactive environmental management. By synthesizing IoT devices and AI models, this research aims to bridge this gap by offering real-time data acquisition, analysis, and prediction of air quality parameters. In doing so, it seeks to provide a comprehensive solution for monitoring various air pollutants and environmental conditions.

Methodologically, this endeavor encompasses both hardware and software components. The hardware configuration involves the utilization of an Arduino Nano microcontroller alongside specific gas sensors (MQ-7, MQ-9, MQ-135) designed to detect carbon monoxide, methane, and ammonia. Additionally, a temperature and humidity sensor (DHT11), a Wi-Fi module (ESP01) for wireless connectivity, and a micro-SD card module for data storage complement the hardware setup. On the software front, programming within the Arduino Integrated Development Environment (IDE), data aggregation and visualization through the ThingSpeak IoT platform, and predictive modeling via Google Colab's AI tools constitute the foundational pillars of this research.

Structured into several chapters, this work embarks on an exploratory journey, beginning with an introduction to Arduino programming and the development environment. It then delves into the realms of artificial intelligence and the Internet of Things, elucidating their respective principles, significance, and operational mechanisms. Subsequent chapters intricately detail the design, implementation, and empirical evaluation of the air quality monitoring system. Through a rigorous discussion of results, including descriptive statistics and comparative analyses of predictive models, the work endeavors to glean insights that pave the way for future advancements in environmental monitoring.

In the first chapter, the thesis embarks on an exploration of the foundational aspects of the Arduino platform. Here, readers are introduced to the features, diverse board types, and fundamental programming principles using the Arduino Integrated Development Environment (IDE). This comprehensive primer lays the groundwork for understanding the hardware and software components essential to the subsequent design of the air quality monitoring system.

Then In the second chapter, the focus shifts towards unraveling the principles underpinning artificial intelligence (AI) and the Internet of Things (IoT). Within this domain, the chapter meticulously elucidates the operational mechanisms of AI and IoT technologies, highlighting their pivotal roles in environmental monitoring.

Moving forward, the third chapter delves into the intricacies of designing and implementing the air quality monitoring system. Here, readers are guided through a systematic exposition of hardware selection and integration, complemented by a detailed elucidation of software development processes. Through this meticulous exploration, readers gain invaluable insights into the construction and operational nuances of the system, culminating in the presentation of the final desired model.

As the work progresses, the last chapter unveils the empirical findings garnered from the air quality monitoring system. Within this realm, descriptive statistics of environmental measurements and a comparative analysis of predictive models take center stage. Methodical evaluation employing metrics facilitates the identification of the most effective AI model for air quality prediction. These empirical insights contribute significantly to the advancement of scientific understanding and inform decision-making in environmental management practices.

In conclusion, this work aspires to not only contribute to the burgeoning field of environmental monitoring but also to provide a robust and scalable solution for real-time air quality assessment. By harnessing the synergistic potential of IoT and AI technologies, it seeks to empower stakeholders with the tools necessary to combat the pervasive threat of air pollution effectively.

Chapter I :
Introduction to Arduino Programming and Development Environment

I.1 Introduction

The Arduino platform has significantly influenced the world of electronics and programming. Since its launch, Arduino has made it easy for hobbyists, students, and professionals to create electronic projects with its combination of easy-to-use hardware and software. Arduino microcontroller boards can read inputs, such as sensors, and control outputs, such as motors and lights.

This chapter aims to introduce Arduino as a whole. We will start by describing the main features of the Arduino platform and its development environment. Then, we will compare the different types of Arduino boards, highlighting their specifications and respective uses. Finally, we will explain the basics of programming in the Arduino Integrated Development Environment (IDE), illustrated with practical examples to better understand its operation and its various applications.

I.2 What is Arduino ?

Arduino is an open-source electronics platform that features both easy-to-use hardware and software. It consists of various microcontroller boards and a development environment for writing, compiling, and uploading code to these boards. [2]

❖ **Key Features:**

- **Microcontroller Boards**: Arduino boards can read inputs from sensors or switches and control outputs such as motors, lights, and other actuators.
- **Input/Output Pins**: These pins can be easily programmed to interact with the external world, enabling a wide range of applications.
- **Programming Language**: The Arduino programming language is based on Wiring and is similar to C/C++. It simplifies coding for microcontrollers by offering functions and libraries that handle low-level hardware details.

❖ **Development Environment:**

- **Arduino IDE**: This integrated development environment is used to write, compile, and upload code to the Arduino boards. It provides a user-friendly interface for programming and debugging. [2]

❖ **Popularity and Applications:**

Arduino has become immensely popular due to its simplicity, versatility, and accessibility. It is widely used by hobbyists, students, educators, and professionals for prototyping projects, including: [2]

- **Simple Experiments**: Such as blinking LED lights.
- **Complex Systems**: Like robotics, home automation, and IoT (Internet of Things) applications.

This platform allows users of all skill levels to create a diverse array of electronic projects, making it a valuable tool in both educational and professional settings.

I.2.1 Types of Arduino Boards – Comparison on Specification and Features:

Choosing the right Arduino board involves carefully evaluating and comparing different specifications across different board types. By thoroughly analyzing these various specifications, users can choose the Arduino board that best matches their project's processing needs, I/O requirements, communication requirements, power limitations, memory footprint, and size/form factor considerations. **Figure I.1** shows the most famous Arduino boards. [2]

Figure I.1: Types of Arduino Boards[2]

Table I.1 provides a comprehensive comparison of various microcontroller boards, detailing their key specifications. It offers a quick overview of the features and capabilities of each board, aiding in the selection process based on project requirements and preferences.

		Uno R3	NANO	Pro-Mini	Leonardo	Micro	Due	Mega2560 Rev3	Zero
Microcontroller		ATmega328P	ATmega328P	ATmega328P	ATmega32u4	ATmega32u4	AT91SAM3X8E	ATmega2560	ATSAMD21G18
FBGA		No	No	No	No	No	No	No	No
	USB connector	USB-B	Mini-B USB	Mini-B USB	Micro USB (USB-B)	Micro USB	Micro USB	USB-B	Micro USB
I/O	Digital only I/O pin	14	14	14	20	20	54	54	20
	Analog input pins	6	8	8	12	12	12	16	6
	Analog output pins	0	0	0	0	0	2	0	1
	PWM Pins	6	6	6	7	7	12	15	10
Communication	UART	Yes	Yes	Yes	Yes	Yes	Yes,4	Yes,4	Yes,2
	I2c	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	SPI	Yes	Yes	Yes	Yes	Yes	Yes (Ext Transceiver)	Yes	Yes
	CAN	No	No	No	No	No	No	No	No
	Bluetooth	No	No	No	No	No	No	No	No
	WIFI	No	No	No	No	No	No	No	No
Power	I/O voltage	5v	5v	5v	5v	5v	3.3v	5v	3.3v
	Input nominal voltage	7-12v	7-12v	7-12v	7-12v	7-12v	7-12v	7-12v	7-12v
	DC Current per I/O	20mA	20mA	20mA	10mA	10mA	9mA/3mA	20mA	7mA
	Power supply connection	Barrel Jack	GPIO header	GPIO header	Barrel Jack	GPIO header	Barrel Jack	Barrel Jack	Barrel Jack
	Battery Powered	No	No	No	No	No	No	No	No
Clock Speed	Main processor	ATmega328P (16MHZ)	ATmega328P (16MHZ)	ATmega328P (16MHZ)	ATmega32u4 (16MHZ)	ATmega32u4 (16MHZ)	84MHZ	16MHZ	48MHZ
RTC		No	No	No	No	No	No	No	No
USB to Serial		ATmega16U2 (16MHZ)	FT232RL	No onboard USB-TTL Converter	Native	Native	ATmega16U2 (16MHZ)	ATmega16U2 (16MHZ)	Native
Memory	Flash	32KB	32KB	32KB	32KB	32KB	512KB	256KB	256KB
	SRAM	2KB	2KB	2KB	2KB	2KB	96KB	6KB	32KB
	EEPROM	1KB	1KB	1KB	1KB	1KB	None	256KB	None
Dimensions	Weight	25g	5g	5g	20g	13g	36g	5g	12g
	Width	53.4mm	18mm	18mm	53.3mm	18mm	53.3mm	18mm	53mm
	Length	68.6mm	45mm	45mm	68.6mm	48mm	101.5mm	45mm	68mm

***Table I.1: Comparison of Arduino Board Types, Specifications and Features**[1]*

I.2.2 Anatomy of an Arduino Board:

While all Arduino boards differ from each other, there are several key components that can be found on practically any Arduino. Let's take a look at **Figure I.2** below: [3]

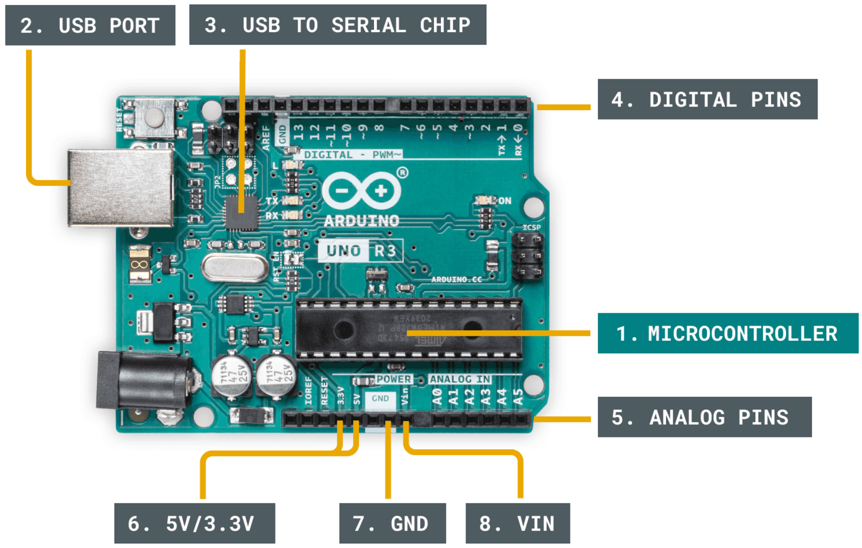


Figure I.2: Key components of an Arduino board. [3]

1. **Microcontroller:** This is the brain of the Arduino. It's a tiny computer designed to execute a specific set of tasks, running the programs you load onto it.
2. **USB Port:** This port is used to connect the Arduino board to a computer, allowing you to upload programs and communicate with the board.
3. **USB to Serial Chip:** This chip translates data between the computer and the on-board microcontroller, enabling the programming of the Arduino from your computer.
4. **Digital Pins:** These pins use digital logic (0 or 1, LOW or HIGH). They are commonly used for tasks like reading switches or turning LEDs on and off.
5. **Analog Pins:** These pins read analog values with a 10-bit resolution (0-1023), allowing for more precise input measurements.
6. **5V / 3.3V Pins:** These pins provide power to external components, supplying either 5 volts or 3.3 volts.
7. **GND (Ground):** Also known as ground, negative, or simply "-", this pin is used to complete a circuit, setting the electrical level at 0 volts.
8. **VIN (Voltage In):** This pin is used to connect external power supplies to the Arduino board. Depending on the specific Arduino model, there may be additional components, but the items listed above are common to most Arduino boards.

I.3 Programming in Arduino IDE 2 Software:

I.3.1 Installing Arduino IDE 2:

Makers, students & professionals have been using the classic Arduino IDE (Integrated Development Environment) ever since Arduino was born.

The Arduino IDE 2 is an improvement of the classic IDE, with increased performance, improved user interface and many new features, such as autocompletion, a built-in debugger and syncing sketches with Arduino Cloud. [4]

In this guide, we will cover the basics of the Arduino IDE 2, where you will find links to more detailed resources on how to use specific features!

- You can download the IDE 2 from the https://www.arduino.cc/en/software
- You can also follow the downloading and installing the Arduino IDE 2 tutorial for more detailed guide on how to install the editor from:

https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2 downloading-and-installing/

The Arduino IDE 2 features a new sidebar, making the most commonly used tools more accessible. The **Figure I.3** shows Interface Arduino IDE 2 software.

Figure I.3: Interface Arduino IDE 2 software[4]

- ❖ **Verify / Upload -** compile and upload your code to your Arduino Board.
- ❖ **Select Board & Port -** detected Arduino boards automatically show up here, along with the port number.

- ❖ **Sketchbook** - here you will find all of your sketches locally stored on your computer. Additionally, you can sync with the Arduino Cloud, and also obtain your sketches from the online environment.
- ❖ **Boards Manager** - browse through Arduino & third-party packages that can be installed.
- ❖ **Library Manager** - browse through thousands of Arduino libraries, made by Arduino & its community.
- ❖ **Debugger** - test and debug programs in real time.
- ❖ **Search** - search for keywords in your code.
- ❖ **Open Serial Monitor** - opens the Serial Monitor tool, as a new tab in the console.

I.3.2 Arduino is programmed with C and C++:

C++ can be considered as a superset of C. It means C++ brings new concepts and elements to C. Basically, C++ can be defined as C with object-oriented implementation (http://en.wikipedia.org/wiki/Object-oriented_programming), which is a higher-level feature. This is a very nice feature that brings and provides new ways of design.

We'll enter together into this concept a bit later in this book but basically, in object-oriented programs, you define structures called **classes** that are a kind of a model, and you create objects called **instances** of those classes, which have their own life at runtime and which respect and inherit the structure of the class from which they came. [5]

I.3.3 Structure:

An Arduino program has two parts:

a. **Setup() Function:** This function is used for preparation and runs once when the program starts. It is always placed at the top of your program. In this section, you typically:
   - Set the modes of the pins (input or output) using **'pinMode()'**.
   - Initialize serial communication with **'Serial.begin()'**.

b. **Loop() Function:** This function is for execution and runs repeatedly after the setup() function. In this section, you write the code that needs to be executed continuously, such as:
   - Reading inputs from sensors or other components.
   - Triggering outputs like LEDs, motors, or other actuators.

By structuring your Arduino program with these two functions, you ensure that your setup configurations are done first, followed by the continuous execution of your main code. [6]

Example:

```
int buttonPin = 3;

void setup() {
  Serial.begin(9600); // Use Serial.begin() instead of beginSerial()
  pinMode(buttonPin, INPUT);
}

void loop() {
  if (digitalRead(buttonPin) == HIGH) {
    Serial.write('H'); // Use Serial.write() instead of serialWrite()
  } else {
    Serial.write('L'); // Use Serial.write() instead of serialWrite()
  }
  delay(1000);
}
```

I.4 Conclusion

Arduino offers a robust and versatile platform for electronic projects. Its combination of easy-to-use hardware and intuitive software makes it accessible to users of all skill levels. The wide range of Arduino boards, each with unique specifications and features, allows for tailored solutions to various project needs. The Arduino IDE enhances the development experience with its comprehensive tools and resources. Understanding the fundamentals of Arduino, from board components to programming structure, empowers users to bring their electronic projects to life efficiently and creatively. Whether for educational purposes or professional development, Arduino stands as a pivotal tool in the realm of electronics and programming.

Chapter II :
Artificial Intelligence & the Internet of Things

II.1 Introduction:

AI and IoT are changing how we live and work. AI makes machines smart, helping in jobs, healthcare, and transport. IoT connects everyday devices to the internet, making data sharing easier. Together, they make life more efficient and innovative.

The objective of this chapter is to present an overview of AI and IoT, exploring their definitions, key impacts, and the challenges they face.

II.2 Artificial intelligence -AI- :

Artificial intelligence (AI) has transcended buzzwords and marketing gimmicks, becoming an indispensable part of our daily lives. Companies now employ AI to create intelligent robots for various applications.

II.2.1 What is AI ?

AI is the study of using vast amounts of data to program intelligent machines. **Figure II.1** offers a conceptual view of AI. By incorporating prior knowledge and experience, these systems can mimic human performance. AI enhances the speed, accuracy, and success with which humans can complete tasks. AI researchers and developers employ sophisticated algorithms and techniques to create autonomous devices. Machine learning and deep learning form the foundations of AI. [7]

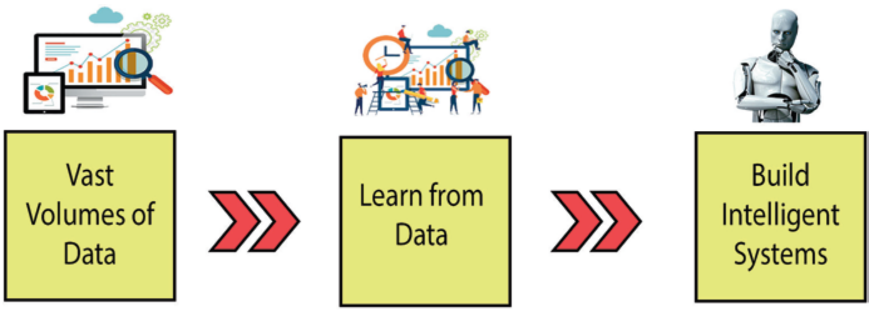


Figure II.1: A conceptual view of artificial intelligence [7]

II.2.2 Why is AI Crucial ?

Artificial Intelligence (AI) is crucial in today's business landscape due to its ability to enhance processes and provide deep insights. AI systems are particularly effective in performing tasks quickly and with high accuracy, especially repetitive and detail-oriented tasks such as verifying fields in legal documents. [7]

- **Key Impacts of AI:**

  - **Efficiency and Accuracy**: AI excels in completing repetitive tasks with minimal errors, improving overall productivity and reducing the likelihood of human error.

  - **Innovative Business Models**:

Uber: Before AI, a company like Uber, which connects riders with taxis via software, would have been unimaginable. Utilizing advanced machine learning algorithms, Uber can predict demand, allowing them to deploy drivers proactively, thereby increasing productivity and accessing new markets. [7]

  - **Market Expansion and User Insights**:

Google: By analyzing user behavior through machine learning, Google has expanded into various online service categories. In 2017, Google's CEO declared the company to be "AI first," highlighting the pivotal role of AI in their strategy (Burns 2017). [7]

- **Competitive Advantage:** The world's largest and most profitable companies are leveraging AI to streamline operations, enhance efficiency, and gain a competitive edge. By adopting AI, these businesses can operate more effectively and explore new opportunities that were previously out of reach. [7]

II.2.3 How Does AI Work ?

As AI gains popularity, some businesses highlight the AI features of their products, often focusing on specific aspects like machine learning. However, AI cannot exist without specialized hardware and software platforms for creating and training machine learning algorithms. Multiple languages, including Python, R, and Java, are commonly used for AI, but none can be considered "the" AI language. [7]

AI typically functions by ingesting vast amounts of labeled training data, processing it to find correlations and patterns, and then using those patterns to predict future outcomes. In this way, an image recognition tool can learn to identify and categorize objects in photographs by analyzing millions of samples, and a chatbot can learn to mimic human conversation by being fed examples of text chats.

AI development prioritizes three cognitive abilities: learning, reasoning, and self-correction.

- **Learning processes:** Algorithms are rules that instruct computers on how to perform something by executing each step in a sequence. This aspect of AI programming focuses on data collection and developing rules for transforming raw data into valuable insights.
- **Reasoning processes:** This aspect of AI programming emphasizes selecting the optimal method to achieve the goal.
- **Self-correction processes:** This aspect of AI development ensures that algorithms consistently produce the most accurate results.

II.2.4 AI Techniques:

Artificial intelligence (AI) engineers need expertise in various fields, including programming and advanced arithmetic, statistics, and data science. This section briefly overviews some core AI techniques (Miao et al. 2021) before discussing typical AI technologies. [7]

II.2.4.1 Classical AI:

The earliest forms of artificial intelligence were symbolic AI, rule-based AI, and good old-fashioned AI (GOFAI). These involved programming computers to follow a series of IF...THEN statements or rules.

For years, "expert systems" based on rule-based AI were developed for various applications like:

- Medical diagnosis
- Credit scoring
- Industrial automation

The foundation of expert systems is knowledge engineering, which requires extensive time, effort, and expertise to successfully elicit and model the knowledge of domain experts.

While an expert system may have hundreds of rules, its reasoning process is generally straightforward. However, as the number of rule interactions grows, expert systems can become increasingly difficult to modify or improve.

The main limitations of these early AI approaches were:

- Their reliance on manually coding IF...THEN rules
- The challenges in capturing and modeling comprehensive expert knowledge
- Difficulty in scaling and maintaining complex rule interactions

This paved the way for more advanced AI techniques like machine learning to overcome these limitations. [7]

II.2.4.2 Machine learning:

Machine Learning (ML) is a method for making predictions without predetermined rules by analyzing large volumes of data to find patterns and build models. Recent advances in ML algorithms have enabled AI breakthroughs like natural language processing, facial recognition, and self-driving cars.

The three main types of ML are:

- Supervised Learning: Uses labeled data (e.g., annotated images) to train models that can be applied to new data (e.g. identifying people in new images).
- Unsupervised Learning: Finds previously unseen patterns and clusters in unlabeled data that can be used to categorize new observations (e.g., recognizing letters/numbers in handwritten text).
- Reinforcement Learning: Continuously refines the model based on feedback/rewards from its performance (e.g., autonomous cars improving collision avoidance through iterative trials).

While ML is a subset of AI, AI systems often combine ML with rule-based or symbolic AI techniques (GOFAI). Many current AI products still require manual input of rules and knowledge from human experts.

Key limitations of current ML approaches:

- ML does not truly "learn" like humans - it depends entirely on human input at each stage (data selection, algorithm design, result interpretation).
- Lauded breakthroughs like object recognition still relied heavily on human-annotated data.
- Complex AI systems like self-driving cars require massive datasets manually labeled by human workers globally.

So, while ML drives many recent AI capabilities, combining it with GOFAI components and large human-annotated datasets is critical for building functional AI systems today. [7]

II.2.4.3 Artificial Neural Networks:

An artificial neural network (ANN) is a type of artificial intelligence that mimics the structure of biological neural networks in animal brains. ANNs have:

- An input layer.
- One or more hidden computational layers.
- An output layer to provide the final result.

ANNs use machine learning techniques like reinforcement learning and backpropagation to adjust the weights between the neurons during training, enabling them to compute outputs for new data inputs.

A famous application of ANNs is Google's AlphaGo, which beat the world's best human Go player in 2016.

The hidden layers in deep neural networks are key to their effectiveness, but also a limitation - their inner workings are opaque and difficult to probe to understand how they arrive at outputs.

This lack of transparency in how ANN and other ML models make decisions that significantly impact humans is an issue many businesses are trying to address through explainable AI techniques.

However, making AI judgments more interpretable also introduces new risks and complexities, as Burt (2019) notes: "producing more knowledge about AI judgments can offer actual benefits, but may also introduce new risks".

So, while ANNs are powerful, their "black box" nature is a double-edged sword that needs to be balanced against the growing importance of explainable and trustworthy AI systems. [7]

II.2.4.4 Deep Learning:

Deep learning refers to multi-layered artificial neural networks (ANNs) with many hidden layers and connections between them. This deep learning approach has been crucial to many recent AI breakthroughs.

Some specific types of deep neural networks include:

- Convolutional Neural Networks (CNNs): Can process multiple data arrays like 2D images as input.
- Recurrent Neural Networks (RNNs): Suitable for tasks like language modeling as they allow bi-directional data flow, process sequential inputs, and learn from previous examples.
- Generative Adversarial Networks (GANs): Involve two neural networks pitted against each other - a generator that creates outputs and a discriminator that evaluates their quality. This adversarial process guides the training cycle.

GANs have been responsible for major achievements, especially in image manipulation tasks such as:

- DeepMind's AlphaZero mastering multiple games using a GAN-based strategy
- Generating photorealistic synthetic portraits of people from training on real photographs

So, while deep learning powers many state-of-the-art AI capabilities today, GANs represent a powerful deep learning technique that has enabled impressive breakthroughs, particularly in image generation and manipulation. [7]

II.3 Internet of Things:

II.3.1 What Is IoT ?

The Internet of Things (IoT) refers to systems of interconnected devices and sensors that collect and process data from the physical world. While the term implies use of the internet, many IoT devices communicate via other networking protocols.

IoT builds upon earlier technologies like pervasive computing, sensor networks, and embedded systems, but takes it further by emphasizing processing of real-time sensor data within the network itself rather than just data collection.

Key characteristics of IoT systems are:

- They are designed for specific applications rather than general internet connectivity
- They account for the dynamics of physical systems through sensors and actuators
- They perform signal processing and time-series data analysis

IoT has been enabled by low-cost MEMS sensors and older, cheap VLSI manufacturing processes sufficient for many IoT nodes.

Power consumption is a critical factor, requiring careful hardware, software, and algorithm design.

Security and safety are key requirements, as IoT merges computational systems with physical systems, necessitating an integrated approach.

In summary, IoT represents domain-specific cyber-physical systems tightly integrating sensing, processing, and actuation rather than a general consumer internet of things. [8]

II.3.2 Applications of IoT Systems:

IoT systems find broad applicability across numerous domains:

- **Industrial Systems:** Sensors monitor industrial processes for quality control and equipment health monitoring, e.g., sensors in electric motors predict impending failures.
- **Smart Buildings:** Sensors detect occupant locations and building state to optimize HVAC and lighting for reduced operating costs and monitor structural integrity.
- **Smart Cities:** Sensors monitor pedestrian and vehicular traffic flows, integrating data from smart buildings.
- **Vehicles:** Networked sensors enable monitoring of vehicle state for improved dynamics, fuel efficiency and emissions reduction.
- **Medical Systems:** Interconnected sensors monitor patients across home, ambulance, clinic and hospital settings.
- **Use Case Categorization:**
  - Sensor Network: Data gathering system for distributed sensors.
  - Alert System: Sensor data analyzed to generate alerts upon meeting criteria.
  - Analysis System: Ongoing analysis of sensor data with periodic reporting.
  - Reactive System: Sensor data analysis triggers actuator responses.

  - Control System: Sensor feedback implemented in control laws driving actuators.
- **Common Non-Functional Requirements:**
  - Event Latency: End-to-end delay from event capture to destination.
  - Event Throughput: Rate of event capture, transport and processing.
  - Event Loss Rate & Buffer Capacity: Tolerance for event losses and buffering needs.
  - Service Latency & Throughput: Performance constraints on event processing services.
  - Reliability & Availability: Dependability metrics for the distributed system components.
  - Service Lifetime: Extended operational duration of the IoT system versus component lifetimes.

These diverse applications and non-functional needs impose corresponding requirements on the IoT system components in areas like processing, networking, power and algorithms. [8]

II.3.3 Security and Privacy:

- **Security Challenges:**

Security is recognized as an essential requirement for IoT systems, yet many deployed IoT systems have glaring security vulnerabilities stemming from: [8]

- Inadequate hardware security features
- Software vulnerabilities and poor design
- Use of default passwords and security misconfigurations

- **Systemic Impacts of Insecure Nodes:**

Insecure IoT nodes create security risks for the overall IoT system due to the typical multi-year lifetimes leading to a large installed base of vulnerable devices.

Moreover, insecure IoT devices can be leveraged to launch attacks on broader Internet infrastructure, as evidenced by IoT-based denial-of-service attacks like the Dyn attack. [8]

- **Privacy Considerations**:

In addition to security, IoT systems must address privacy concerns:

- Protecting user data from theft/exposure
- Avoiding leakage of private information from less-sensitive data
- Requiring privacy protections at application, network and device levels

Comprehensively addressing both security and privacy from device capabilities to system architectures is crucial for trustworthy and reliable IoT system operation. [8]

II.4 Conclusion:

Artificial Intelligence (AI) and the Internet of Things (IoT) continue to revolutionize our lives by improving efficiency and connectivity. Despite the challenges, especially in terms of security, their potential to positively transform our world is immense. Collaboration and innovation will continue to drive these technologies to new heights.

Chapter III :
System design and implementation

III.1 Introduction :

The intelligent gas monitoring system utilizes advanced hardware and software to detect and analyze various gases. The hardware components include sensors, microcontrollers, and power supplies, while the software components involve data processing and IoT integration. This combination ensures precise data collection and real-time monitoring, improving safety and efficiency.

The objective of this chapter is to present an overview of the system design and implementation, exploring the hardware and software components, their interactions, and the overall functionality of the gas monitoring system.

III.2 Basic components of the system:

For this intelligent gas monitoring system, we have carefully selected a range of components that work in synergy to enable accurate data collection, processing, and analysis. These components are divided into:

❖ **Hardware aspect**	❖ **Software aspect**
• **Arduino nano with cable** • **MB102 Breadboard 3.3V/5V Power Supply with cable** • **MQ-7** • **MQ-9** • **MQ135** • **DHT11** • **ESP01** • **Micro SD Card Module**	• **Arduino IDE 2 Software** • **ThingSpeak IoT** • **Google Colab AI**

III.2.1 Hardware aspect:

❖ Arduino Nano with cable:

Arduino Nano is a superficial mount little microcontroller breadboard embedded model with integrated Micro USB Port.

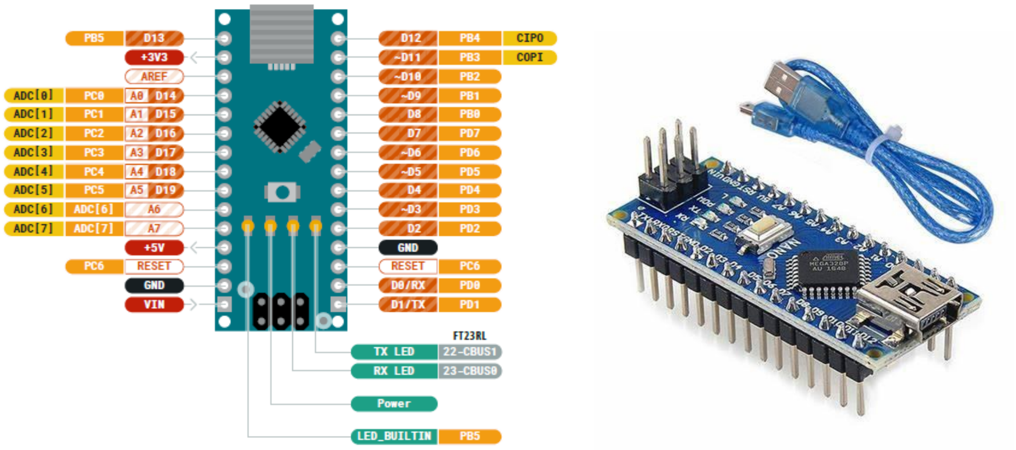


Figure III.1: Arduino Nano [4]

It is a tiny, full-fledged microcontroller, breadboard friendly, and cost-effective device, it has everything like same as Die/Due/Uno board has (electrically) with extra digital and analog I/O pins and onboard +5V AREF jumper.

(The properties are mentioned in *Table I.1* in the chapter)

❖ Measurement of GAS using MQ-Series (Gas sensor Module):

The MQ series of gas measuring device uses a small heater inside with an electrochemical sensor. They are delicate for a range of gasses and are used in the house at room temperature. The output is an analog signal and can be read with an analog input of the Arduino.

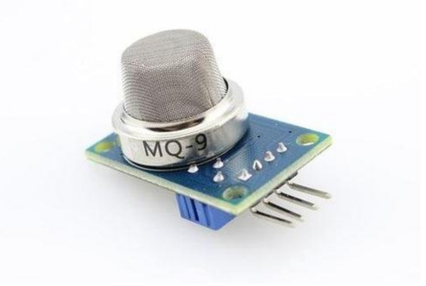
Here are MQ series of gas sensors used in this system:

- MQ-7: They are used in gas detecting equipment for carbon monoxide (CO) in family and industry or car.[9]

Figure III.2: MQ-7 for carbon monoxide (CO)

- MQ-9: is gas sensor has high sensitivity to Carbon Monoxide, Methane and LPG. The sensor could be used to detect different gases contains CO and combustible gases, it is with low cost and suitable for different application.[10]

Figure III.3: MQ-9 Carbon Monoxide, Methane and LPG Gas Sensor



- MQ-135: They are used in air quality control equipment's for buildings/offices, are suitable for detecting of NH3, NOx, alcohol, Benzene, smoke, CO2, etc.[11]

Figure III.4: MQ-135 for Air Quality

<u>Common characteristics across these semiconductor gas sensors:</u>[9-11]

a. Structure and Configuration:
   - All composed of a micro alumina (Al2O3) ceramic tube
   - Contain a sensitive tin oxide (SnO2) sensing layer
   - Have a measuring electrode and heater coil fixed into a plastic and stainless-steel mesh casing
b. Operating Conditions:
   - Operate at a low circuit voltage around 5V DC
   - Require a separate heating voltage (typically 5V for high heat and 1.4-1.5V for low heat)
   - Have specific heating times for high and low operation (typically a 60-90 second cycle)
   - Work best at certain temperature and relative humidity ranges (typically 20°C and 65% RH)

c. Operating Principle :

- Rely on changes in their surface resistance (sensing resistance Rs) when exposed to target gases
- Have a resistance ratio to clean air (Rs/Ro) that increases with higher gas concentrations
- Resistance measured through an electronic circuit containing a load resistance RL

d. Applications:

- Designed for use in industrial, residential, and portable gas detection devices
- Target specific gases

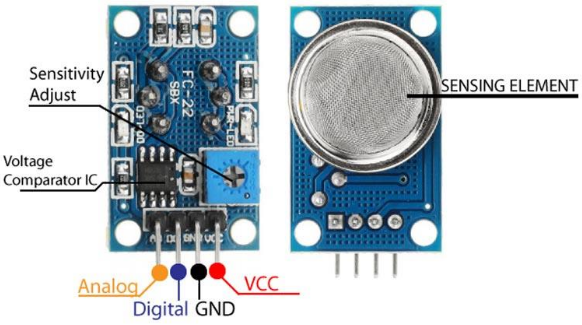


Figure III.5: MQ Gas Sensor Module Pinout [12]

❖ **Temperature and Humidity Measurement using DHT11 Sensor:**

The DHT11 humidity and temperature sensor make it really easy to add humidity and temperature data to your DIY electronics projects. It's perfect for remote weather stations, home environmental control systems, and farm or garden monitoring systems. [13]

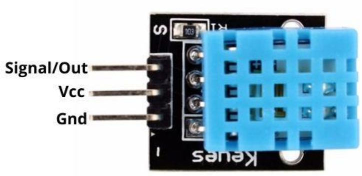


Figure III.6: DHT11 Sensor [13]

Here are the ranges and accuracy of the DHT11: [13]

- Humidity Range: 20-90% RH
- Humidity Accuracy: ±5% RH
- Temperature Range: 0-50 °C
- Temperature Accuracy: ±2% °C
- Operating Voltage: 3V to 5.5V

❖ **Espressif ESP8266 Serial ESP-01 Wi-Fi Module:**

The ESP-01 is a compact wireless module based on the ESP8266EX Wi-Fi SoC from Espressif. It is a low-cost, compact serial-to-Wi-Fi solution for basic wireless integration into embedded projects and IoT product development. [14]

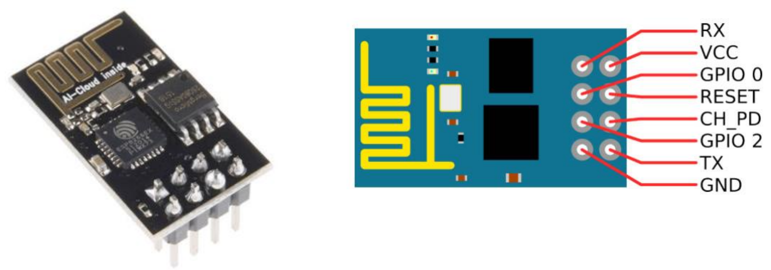


Figure III.7: ESP-01 Wi-Fi Module [15]

Specifications [14]

- Power Supply: +3.3V only
- Current Consumption: 100mA
- I/O Voltage: 3.6V (max)
- I/O source current: 12mA (max)
- Built-in low power 32-bit MCU @ 80MHz
- 512kB Flash Memory
- 8Mbit external QSPI flash memory (1MByte)
- 32-bit Tensilica Xtensa LX106 CPU running 80MHz
- 3.3V supply (current can spike 300mA+, depending on mode)
- PCB-trace antenna
- 2 x 4 dual-in-line pinout
- Dimensions: 14.3 x 24.8mm
- weight: 1.5g

❖ **Micro SD Card Module:**

The module (Micro-SD Card Adapter) is a Micro SD card reader module, and the SPI interface via the file system driver, microcontroller system to complete the Micro-SD card read and write files. Arduino users can directly use the Arduino IDE comes with an SD card to complete the library card initialization and read-write. [16]

Figure III.8: Micro SD Card Adapter Module [16]

Features: [16]

- Supports micro-SD card (<=2G), micro SDHC card (<=32G) (high-speed card)
- Level conversion circuit board that can interface level is 5V or 3.3V
- Power supply is 4.5V ~ 5.5V, 3.3V voltage regulator circuit board
- Communication interface is a standard SPI interface
- 4 M2 screw positioning holes for easy installation
- Size: 4.1 x 2.4cm

❖ **MB102 Breadboard 3.3V/5V Power Supply with cable USB (Male-Male):**

A 3.3V and 5V breadboard power supply module with a series diode and polarity reversal protection. The module can accept an input voltage ranging from 6.5V to 12V and produce 3.3V and 5V outputs. This module is essential for experimenters who need to test or prototype electronic circuits on breadboards or perforated/Vero boards. [17]

Figure III.9: breadboard power supply module with cable USB (Male-Male)

Features: [17]

- Input voltage: 6.5-12 V (DC) or 5V USB power supply
- Output voltage: 3.3V and 5V, switchable
- Maximum output current: <700 mA
- External input voltage ON/OFF switch
- Independent control of upper and lower breadboard power rails: Can switch to 0V, 3.3V, or 5V using jumpers on any rail
- On-board two groups of 3.3V and 5V DC output plug pins for convenient external lead use
- USB device connector onboard for power output to external devices
- Size: 5.3cm x 3.5cm

❖ **Breadboard:**

The breadboard is the most critical component in any project. This board allows the user to build circuits. It's like a patch panel, with rows of holes that allow you to connect wires and components together. This eliminates the need for soldering of components. [18]

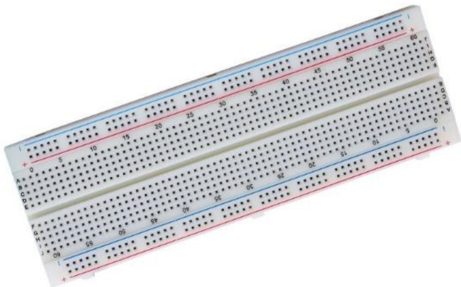


Figure III.10: breadboard

Features and Specifications: [18]

- 2 Distribution Strips, 200 tie-points
- 630 tie-points in IC/ circuit areas
- ABS plastic with color legend
- Dimension: 6.5x4.4x0.3 inch
- Hole/Pitch Style: Square wire holes (2.54mm)
- ABS heat Distortion Temperature: 84° C (183° F)
- Rating: 300/3 to 5Amps
- Insulation Resistance: 500MΩ / DC500V
- Withstanding Voltage: 1,000V AC / 1 minute
- Insertion Wire Size: 21 to 26 AWG wire

❖ **Jumper wires**:

Jumper wires are simple wires with connector pins or metal tip ends that allow them to be used for making temporary connections between points without soldering. They are primarily used with breadboards and other prototyping tools. Their main purpose is to provide an easy way to change, modify or reconfigure a circuit as needed during the prototype and experimentation phase.[19]

***Figure III.10:* Jumper wires**

III.2.2 Software aspect:

☒ **Arduino IDE 2 Software**: Mentioned in Chapter I

☒ **ThingSpeak:**

ThingSpeak is an **IoT analytics platform service** that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak from your devices, create instant visualization of live data, and send alerts. It's particularly useful for IoT projects that require data collection and analysis with advanced tools like MATLAB.

Figure III.10: ThingSpeak IoT logo [20]

the features and applications of ThingSpeak: [20]

- **Data Collection**: Send sensor data privately to the cloud.
- **Analysis**: Analyze and visualize your data with MATLAB.
- **Action**: Trigger reactions based on the data received.
- **Integration**: ThingSpeak works with various devices and services like Arduino®, ESP8266, Raspberry Pi™, and more.
- **Applications**: It's used for environmental monitoring, energy monitoring, smart farming, and other IoT services.

☒ Colab AI:

Colab AI refers to various AI-powered tools and functionalities that are available on Google Colab, a platform that allows users to write and execute code in the cloud. It's particularly popular for machine learning and data science projects because it provides free access to GPUs and TPUs, which can significantly speed up computations.

Figure III.11: Colab AI logo

some highlights of what you can do with Colab AI:

- **KoboldAI**: This is an AI-based text generation tool that you can use on Google Colab. It's suitable for writing stories, blog posts, playing text adventure games, or even assisting with coding tasks. However, it's important to verify the information provided by the AI as it can sometimes generate inaccurate content**.** [21]
- **Hugging Face Integration**: Google Colab now allows users to open and execute notebooks hosted on the Hugging Face Hub. This integration provides a seamless experience for exploring datasets, training models, and building demos directly in Colab .[22]
- **Coding Support**: Similar to GitHub Copilot, Colab AI can offer coding support, acting as a pair programming partner to help you with your coding projects .[23]

These tools make Google Colab a versatile environment for anyone working in fields that require heavy computational resources, from students to researchers in AI. Remember to keep the Colab tab open while working to avoid session timeouts and to respond to any captchas promptly to maintain your session.

III.3 Step-by-Step Guide:

a. Setting Up the Arduino IDE:

- Download and install Arduino IDE 2.
- Connect the Arduino Nano to your computer using the USB cable.
- Open Arduino IDE and select the correct board and port under the Tools menu.

b. Wiring the Components:

- Place the Arduino Nano on the breadboard.
- Connect the MB102 Breadboard Power Supply to the breadboard and power it via the USB cable.
- Wire the sensors (MQ-7, MQ-9, MQ135, DHT11) to the Arduino Nano according to their respective datasheets.
- Ensure you provide the correct power supply voltage (3.3V or 5V).
- Connect the sensor outputs to the analog or digital pins on the Arduino Nano.
- Connect the ESP01 to the Arduino Nano (note that ESP01 requires a 3.3V power supply).
- Connect the Micro SD Card Module to the Arduino Nano

To set up the system using an Arduino Nano, MB102 Breadboard Power Supply, and various, you can follow these connections in **Table III.1** to get the first model shown as **Figure III.11** :

		Arduino nano	**MB102 Breadboard 3.3V/5V Power Supply**
MQ-7	VCC	5V	\
	GND	GND	\
	A0	A0	\
MQ-9	VCC	5V	\
	GND	GND	\
	A0	A1	\
MQ135	VCC	5V	\
	GND	GND	\
	A0	A2	\
DHT11	(+)	5V	\
	Out	D7	\
	(-)	GND	\
ESP01	RX	D3	\
	$GPIO_0$	\	\
	$GPIO_2$	\	\
	GND	GND	3v3(-)
	3v3	\	3v3(+)
	RST	\	\
	CH_PD	\	3v3(+)
	TX	D2	\
Micro SD Card Module	MOSI	11	\
	MISO	12	\
	SCK	13	\
	CS	10	\
	VCC	\	3v3(+)
	GND	\	3v3(-)

Table III.1: wiring/connection table for various components

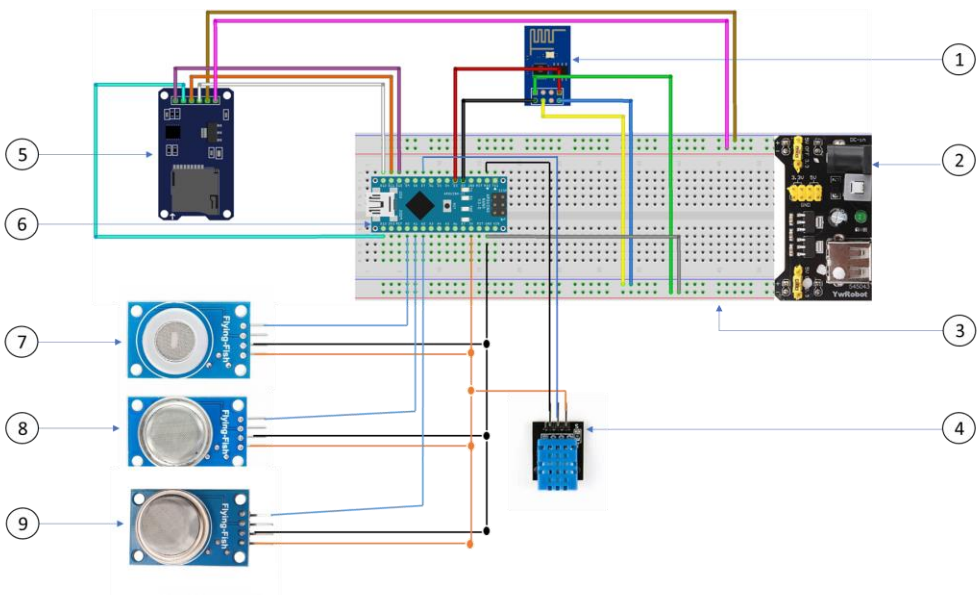


Figure III.11: The first model of the system (Fritzing platforme)

1. ***ESP01***
2. ***3.3V/5V Power Supply***
3. ***Breadboard***
4. ***DHT11***
5. ***Micro SD Card Module***
6. ***Arduino nano***
7. ***MQ-7***
8. ***MQ-9***
9. ***MQ135***

c. Setting Up ThingSpeak:

☒ **Create a ThingSpeak Account:**

- Go to ThingSpeak and sign up for a free account.
- Log in to your ThingSpeak account.

☒ **Create a New Channel:**

- Navigate to the Channels tab and click on "New Channel".
- Fill in the details for your channel, such as name and description, and create fields for the data you will collect (e.g., Temperature, Humidity, CO-Concentration …).
- Save the channel settings.

d. Writing the Arduino Code:

☒ Main Includes:

– The 'SoftwareSerial.h' and 'DHT.h' libraries are included, which are required for using the modem and the humidity/temperature sensor.
– The 'SD.h' library is included to facilitate communication with the Micro SD card.
– The pin numbers for each sensor and modem are defined.
– The calibration values for the gas sensors are defined to calculate their concentrations.
– The Wi-Fi connection details and ThingSpeak credentials are defined.

☒ Gas Concentration Calculation Functions:

– There are three separate functions to calculate the concentration of each gas using a specific equation for each sensor.
– These functions take the raw value from the sensor and the gas-specific calibration values to calculate the concentration.

- Initial Setup:
  - The serial communication is set up for the Arduino and the modem.
  - The humidity and temperature sensor is started.
  - AT commands are sent to the modem to initialize the Wi-Fi connection.
- Main Loop:
  - The raw values are read from the gas sensors.
  - The gas concentrations are calculated using the respective sensor functions.
  - The humidity and temperature are read from the DHT11 sensor.
  - The gas concentrations are printed to the serial monitor.
  - The gas, humidity, and temperature data are prepared to be sent to ThingSpeak.
  - AT commands are sent to the modem to establish TCP connections with ThingSpeak and send the data.
  - The connections are closed after sending the data.
  - After sending data, The 'logData()' function is called to log the same data to the SD card. This ensures that sensor data is stored locally on the SD card in addition to being transmitted over Wi-Fi to ThingSpeak.
- Send Command Function:
  - This is a helper function to send AT commands to the modem and verify the correct response.
  - It is used to control the process of establishing connections and sending data to ThingSpeak.

Overall, this code reads data from various sensors, calculates the gas concentrations using specific equations, and then sends this data to an internet service (ThingSpeak) over a Wi-Fi connection. The uploaded data can be used for displaying, analysis, or setting up alerts if certain concentration levels are exceeded.

NOTE: By adding Micro SD card logging functionality, the system becomes more reliable and resilient, as it provides a backup mechanism for storing sensor data locally in case of network disruptions. This ensures data integrity and continuity of operation, even in adverse conditions**.**

e. Visualize Data on ThingSpeak:

- Go to your ThingSpeak channel.
- Click on the "Private View" or "Public View" tab to see real-time graphs of your data.
- Customize the visualizations as needed to suit your project's requirements.
- Export the logged data

f. Predicting Outcomes Using AI Models in Google Colab:

This guide demonstrates how to predict outcomes using various AI models (LSTM, Random Forest, Multiple Linear Regression, and Support Vector Machine) in Google Colab. The key steps include data preparation, model training, prediction, and evaluation. Here's a breakdown of the process:

- **Set Up Google Colab and Prepare Data:**
  - Open Google Colab and Sign in with your Google account.
  - load your dataset.
  - Select relevant columns, normalize the data, and define the sequence length.
  - Create sequences from the data and split them into training and testing sets.
  - Reshape data for compatibility with different models.
- **Build and Train Models:**

Here's a brief overview of each model used:[24]

Model	Description
Artificial Neural Network (ANN)	A computational model inspired by the human brain's network of neurons. It's good for capturing nonlinear relationships in data. The Artificial Neural Network (ANN) utilized in this study is a fully connected feedforward neural network, a fundamental architecture in machine learning. It comprises an input layer, two hidden layers, and an output layer. The hidden layers are designed with 64 and 32 neurons respectively, employing the Rectified Linear Unit (ReLU) activation function to introduce non-linearity. The model is optimized using the Adam optimizer, which dynamically adjusts learning rates to improve convergence. Training is conducted over 50 epochs, using the Mean Squared Error (MSE) loss function to quantify prediction error, making it suitable for regression tasks.
k-Nearest Neighbors (k-NN)	A simple algorithm that classifies a data point based on how its neighbors are classified. It's easy to implement and understand. The k-Nearest Neighbors (k-NN) algorithm implemented is an instance-based learning model, characterized by its simplicity and effectiveness for classification and regression. The model operates by storing the training dataset and predicting outcomes based on the majority class or average of the k-nearest neighbors, with k set to 5. This method leverages distance metrics to identify the closest data points, providing a straightforward but powerful means of making predictions based on local information.

Convolutional Neural Network (CNN)	Particularly effective for image and video recognition, CNNs can also be applied to time series data for air quality prediction. The Convolutional Neural Network (CNN) applied in this research captures spatial hierarchies through its layered architecture. It features a convolutional layer with 64 filters and a kernel size of 2, which effectively extracts features from the input data. This is followed by a flattening layer and dense layers, with the final dense layer containing 50 neurons. The CNN uses the ReLU activation function and the Adam optimizer, training over 50 epochs with MSE as the loss function, making it well-suited for tasks involving structured grid data such as images.
CNN-LSTM	A combination of CNN and LSTM (Long Short-Term Memory) networks, useful for capturing spatial and temporal patterns, respectively. The CNN-LSTM hybrid model integrates convolutional layers with Long Short-Term Memory (LSTM) units to leverage both spatial and temporal dependencies. The convolutional layer, configured with 64 filters and a kernel size of 2, precedes an LSTM layer with 50 units. This combination allows the model to capture intricate patterns within the data. Training is conducted over 50 epochs using the Adam optimizer and MSE loss function, providing a robust approach for tasks involving sequential and spatial data.
CNN-LSTM with Bayesian Optimization	This is an advanced model that combines CNN, LSTM, and BOA (Bayesian Optimization Algorithm) for optimizing the model parameters. The CNN-LSTM model with Bayesian Optimization further enhances model performance by optimizing hyperparameters. The convolutional layer's filter count ranges from 32 to 128, and the kernel size ranges from 1 to 5, while the LSTM layer's units range from 50 to 200. The optimization process spans 20 epochs for each hyperparameter set, with three iterations following two initial points. This method systematically explores the hyperparameter space to identify the optimal configuration, thereby improving predictive accuracy and model efficiency.
Long Short-Term Memory (LSTM)	A type of recurrent neural network (RNN) that can learn order dependence in sequence prediction problems. The Long Short-Term Memory (LSTM) network, a specialized form of Recurrent Neural Network (RNN), is designed to handle sequence data with long-term dependencies. It features an LSTM layer with 50 units, followed by a dense layer. The ReLU activation function is used within the LSTM units, and the model is optimized using Adam. Trained over 50 epochs with MSE as the loss function, this model excels in tasks involving temporal sequences, such as time series forecasting.
Random Forest	An ensemble learning method that operates by constructing multiple decision trees. It’s robust to overfitting and can handle various types of input data. The Random Forest model, an ensemble learning technique, constructs multiple decision trees during training to improve predictive performance. The model in this study comprises 100 trees, each trained on a random subset of the data. This approach reduces overfitting and enhances generalization by averaging the predictions from individual trees, making it a powerful tool for both classification and regression tasks.
Multiple Linear Regression	A statistical technique that models the relationship between two or more features and a response by fitting a linear equation to observed data. The Multiple Linear Regression (MLR) model predicts the target variable based on a linear combination of multiple predictor variables. It fits a linear equation to the data, minimizing the residual sum of squares to find the best-fitting line. MLR is a simple yet effective method for regression tasks, providing insights into the relationships between independent and dependent variables.

Support Vector Machine (SVM)	A powerful classifier that works well on a wide range of classification problems, including those with complex boundaries. The Support Vector Machine (SVM) is a supervised learning algorithm capable of performing classification and regression tasks. SVM constructs a hyperplane or set of hyperplanes in a high-dimensional space, which can be used for classification or regression tasks. The algorithm aims to find the hyperplane that best separates the data points into different classes or predicts continuous values with the maximum margin of separation.

Table III.2: Model Descriptions and Capabilities

- **Make Predictions:**
  - Use each trained model to make predictions on the test data.
  - Invert the scaling of predictions and actual values for proper comparison.

- **Plot Predictions vs Actual Values:**
  - Visualize predictions against actual values for each model.
  - Combine plots for all models to compare their performance visually.

- **Evaluate Model Performance**: Calculate performance metrics for each model:
  - Mean Absolute Error (MAE)
  - Mean Absolute Percentage Error (MAPE)
  - R-squared
  - Mean Squared Log Error (MSLE)
  - Root Mean Squared Error (RMSE)
- **Summarize Metrics:**
  - Create a DataFrame to store and display the performance metrics of all models.
  - Compare metrics to determine which model performs best.

III.4 How the system works ?

The Schematic illustrates the components of an intelligent air quality monitoring system and shows the division between the hardware aspect and the software aspect as follows:

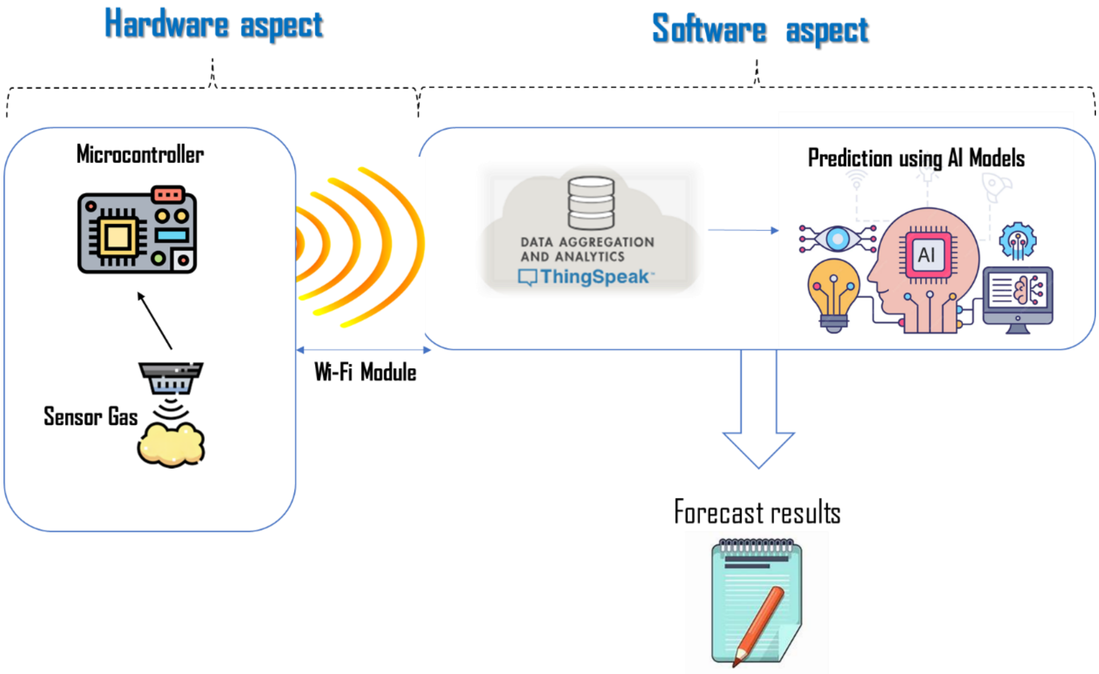


Figure III.12: Schematic overview of the smart system to study air quality

Hardware Aspect:

- **Sensors:** Sensors are used to collect air quality data from the environment, such as pollutant levels.
- **Processing Unit:** This unit receives the sensor data and performs initial processing.
- **MicroSD Card Module**: used in data logging
- **Wi-Fi Module**: This module enables wireless communication, allowing the data collected by the sensors to be transmitted to the cloud

Software Aspect:

- **Data Aggregation and Analytics:** The data is sent to the ThingSpeak platform for aggregation and analysis.
- **Prediction using AI Models:** AI models are used to analyze the aggregated data and predict future air quality levels.
- **Output:** The results are outputted in the form of reports or alerts, which can be used to make appropriate decisions.

III.5 The final desired model:

This final form of the system showen as **Figure III.12** is still being designed and tested to ensure optimal performance and reliability. Various aspects such as sensor selection, data processing algorithms, communication protocols, and user interface design are being carefully considered and refined to meet the specific requirements and objectives of the air quality monitoring system.

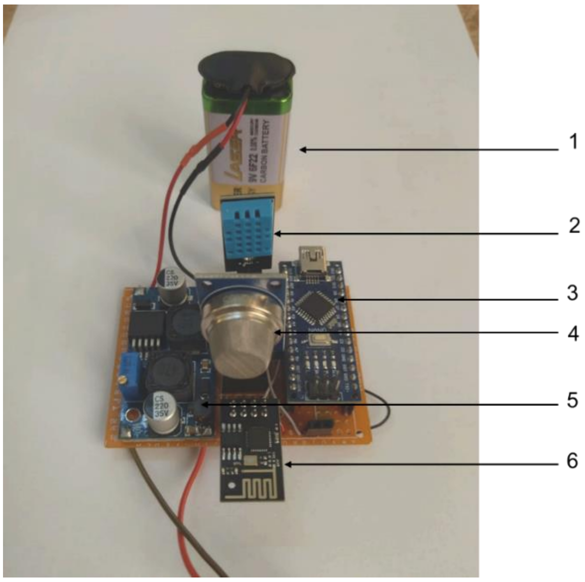


Figure III.12: The final desired model for an intelligent air quality monitoring system.

1. External feeding
2. DHT11 temperature and humidity sensor
3. Arduino Nano
4. MQ gas sensor
5. Step-up power module
6. ESP01 module

Additionally, extensive testing is being conducted to validate the accuracy and consistency of sensor readings, the effectiveness of calibration procedures, and the robustness of the overall system in different environmental conditions. This testing phase involves simulated scenarios as well as real-world deployments to assess the system's performance under various operational conditions.

Overall, the ongoing design and testing phase is essential for refining the system and ensuring that it achieves its objectives of providing accurate, reliable, and actionable air quality data for monitoring and analysis purposes. As a recent development, we have integrated an SD module into the system to mitigate the impact of internet interruptions. This addition allows for the local storage of sensor data in situations where the internet connection is disrupted, ensuring continuity of data collection and preventing any loss of critical information.

III.6 Conclusion :

The intelligent gas monitoring system integrates carefully selected hardware and software components to achieve accurate gas detection and analysis. The use of sensors, microcontrollers, and IoT platforms enables efficient data collection and real-time monitoring. This system enhances safety and operational efficiency by providing reliable and timely information about gas levels. The comprehensive design and implementation discussed in this chapter lay the foundation for further advancements and applications in gas monitoring technology.

Chapter IV : Discussion of Results

IV.1 Introduction :

The predictive analytics system leverages advanced machine learning models to forecast environmental parameters. The system includes various models such as Random Forest, Multiple Linear Regression, and CNN-LSTM, which are evaluated based on metrics like MAE, MAPE, and R-squared. This combination of models ensures accurate predictions and robust analysis, contributing to better environmental management and policy-making.

The objective of this chapter is to discuss the performance evaluation of these predictive models, examining their accuracy, reliability, and implications for environmental data analysis. This exploration highlights the strengths and weaknesses of each model and their suitability for different predictive tasks.

IV.2 Descriptive Statistics for Environmental Measurements:

The **Table IV.1** provides a comprehensive statistical overview of environmental data, showing the distribution, central tendency (mean and median), and variability (standard deviation) of each variable.

	CO (PPM)	CH4 (PPM)	NH3 (PPM)	TEMPERATURE (°C)	HUMIDITY (% RH)
COUNT	31317.0	18885.0	40873.0	40256.0	40256.0
MEAN	609.45	10637.92	105.47	29.91	29.41
STD	118.36	4389.36	17.06	3.33	10.67
MIN	149.8	1091.05	23.85	21.4	0.0
25%	531.25	7209.12	98.27	27.4	20.0
50%	597.47	9822.92	106.76	29.6	26.8
75%	691.3	14666.67	116.67	32.5	36.6
MAX	1917.31	34687.5	351.44	39.4	90.8
KS STATISTIC	0	0	0	0	0
P-VALUE	1	1	1	1	1

Table IV.1: Descriptive Statistics for Environmental Measurements: CO, CH4, NH3, Temperature, and Humidity

Table (IV.1) analysis:

This information in the table can be used to understand the general behavior of these variables and is useful for further analysis such as detecting trends, identifying anomalies, and exploring relationships between variables.

- **Count:** The number of observations or measurements recorded for each variable.

- **Mean:** The average value of the measurements for each variable.
- **Standard Deviation (std):** A measure of the amount of variation or dispersion of the measurements.
- **Minimum (min):** The lowest recorded value for each variable.
- **25th Percentile (25%):** The value below which 25% of the observations fall.
- **Median (50%):** The middle value when the measurements are ordered from lowest to highest. This is also known as the 50th percentile.
- **75th Percentile (75%):** The value below which 75% of the observations fall.
- **Maximum (max):** The highest recorded value for each variable.
- **KS Statistic:** This is the Kolmogorov-Smirnov statistic, which tests the hypothesis that the sample comes from a specified distribution.
- **P-Value:** The p-value associated with the KS statistic indicates the probability of observing the given result, or something more extreme, assuming the null hypothesis is true. In this case, a p-value of 1 suggests that the sample distributions do not significantly differ from the specified distributions.

Key Observations:

- ☒ CO and NH3:
  - Both gases exhibit a relatively normal distribution, as evidenced by the mean and median values being close together.
  - The standard deviations are relatively small compared to their means, suggesting that most measurements are clustered around the mean, indicating less variability.
- ☒ CH4:
  - Methane shows higher variability, indicated by a larger standard deviation relative to the mean.
  - The wide range between the minimum and maximum values suggests significant fluctuations in methane levels, possibly due to varied sources or environmental conditions.
- ☒ Temperature and Humidity:
  - Temperature measurements are more consistent, with a smaller standard deviation, implying values are tightly clustered around the mean.
  - Humidity exhibits a broader spread, as indicated by a higher standard deviation, suggesting more variability in humidity levels.

- ☒ Extremes:
  - The minimum and maximum values for each variable highlight the presence of potential outliers or significant environmental variations.
  - For example, the minimum humidity of 0% is notably extreme and indicate very dry conditions.
- ☒ Statistical Tests**:** The KS Statistic and p-value indicate that the distributions of these variables do not deviate significantly from the expected distributions.

Implications for Further Analysis:

- **Trend Detection:** Longitudinal analysis could reveal trends over time for each variable, helping to understand seasonal or longer-term changes.
- **Anomaly Identification:** Identifying outliers and understanding their causes could be crucial for environmental monitoring and management.
- **Correlational Studies:** Exploring correlations between these variables can provide insights into how they interact, such as the relationship between temperature and humidity or the impact of temperature on gas concentrations

IV.3 Comparative Analysis of Predictive Models:

The primary goal of this analysis is to evaluate the accuracy and reliability of different predictive models in forecasting CO concentration. By comparing these models, we aim to identify the most effective model for environmental data analysis, thereby enhancing climate prediction strategies and contributing to better environmental management and policy-making.

In each of **Figure IV.1** to **Figure IV.9** it explains Predictions and Actual of several models for CO concentration where :

(A) images provides a detailed time series comparison between the actual CO concentration values and the predicted values for each model over a specific period. Each model's predictions are represented in purple, while the actual values are in green. This allows for precise evaluation of each model's performance over time.

(B) images demonstrates the out-of-sample prediction performance of several models for CO concentration (After deleting the periods that do not contain data).

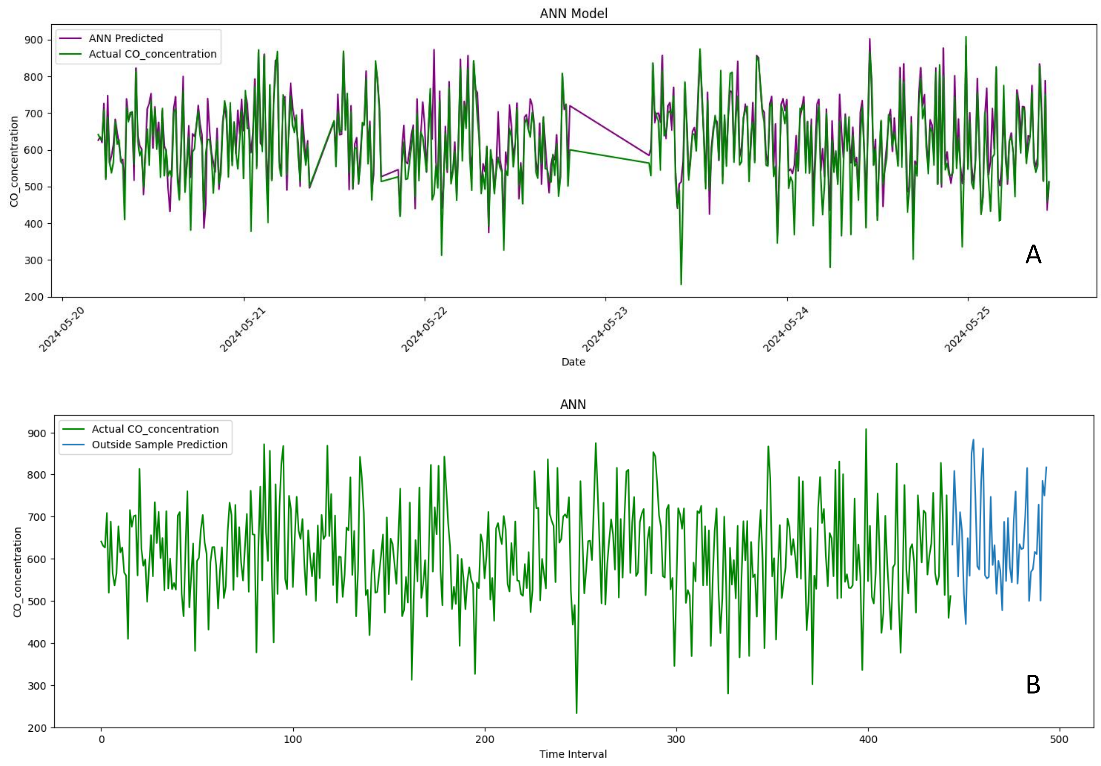


Figure IV.1: ANN Predictions and Actual CO Concentrations

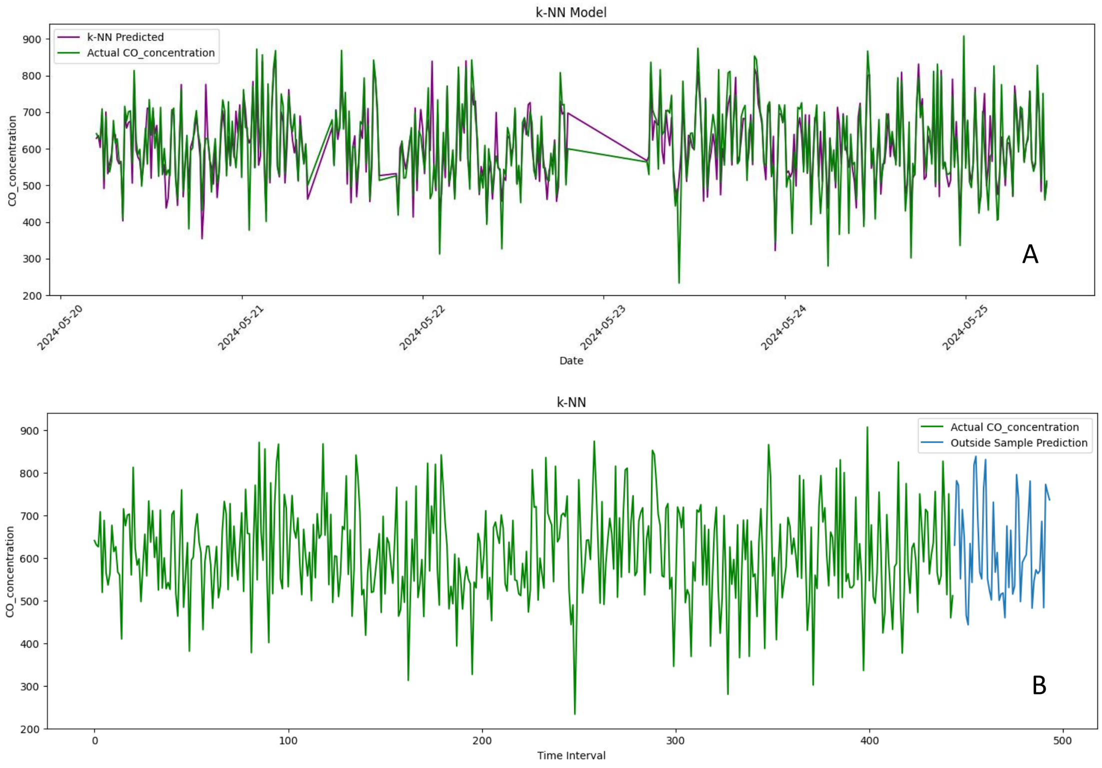


Figure IV.2: K-NN Predictions and Actual CO Concentrations

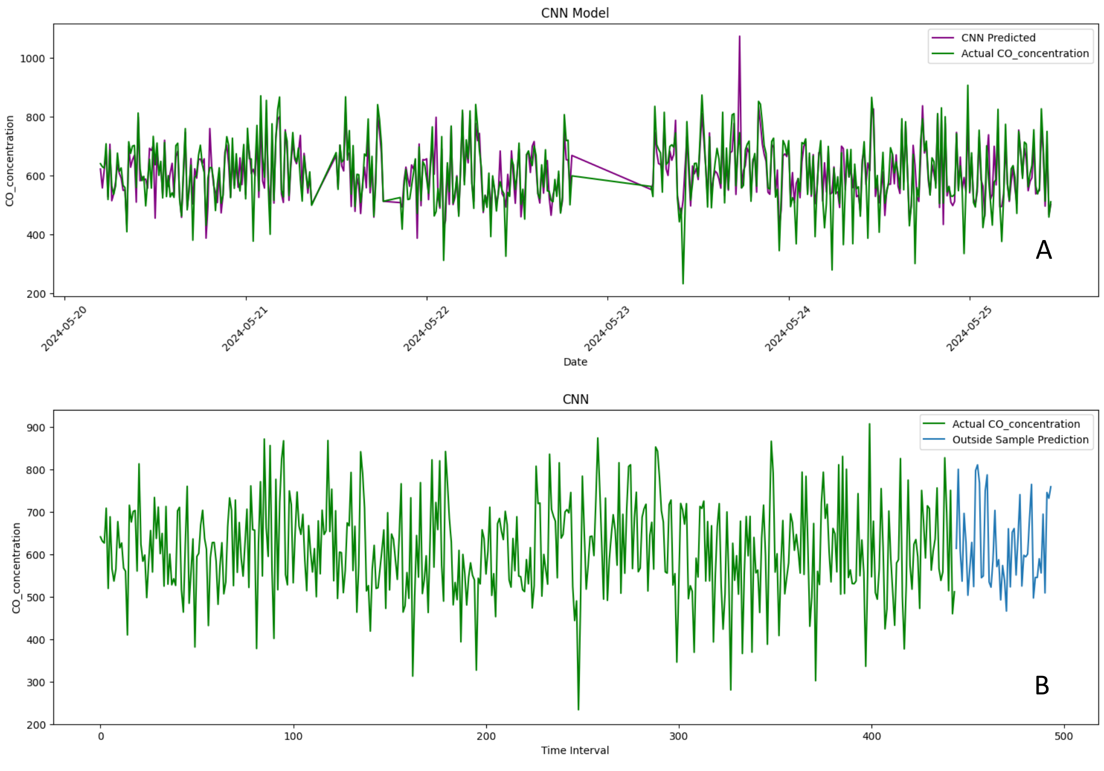


Figure IV.3: CNN Predictions and Actual CO Concentrations

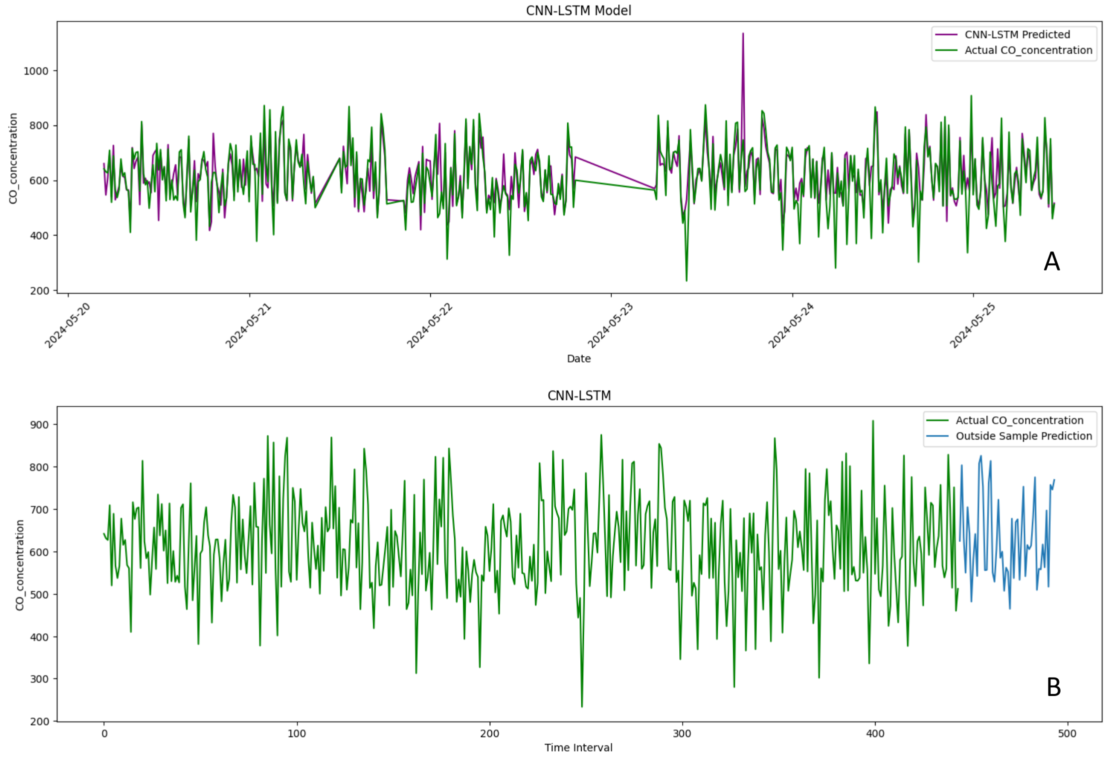


Figure IV.4: CNN-LTSM Predictions and Actual CO Concentrations

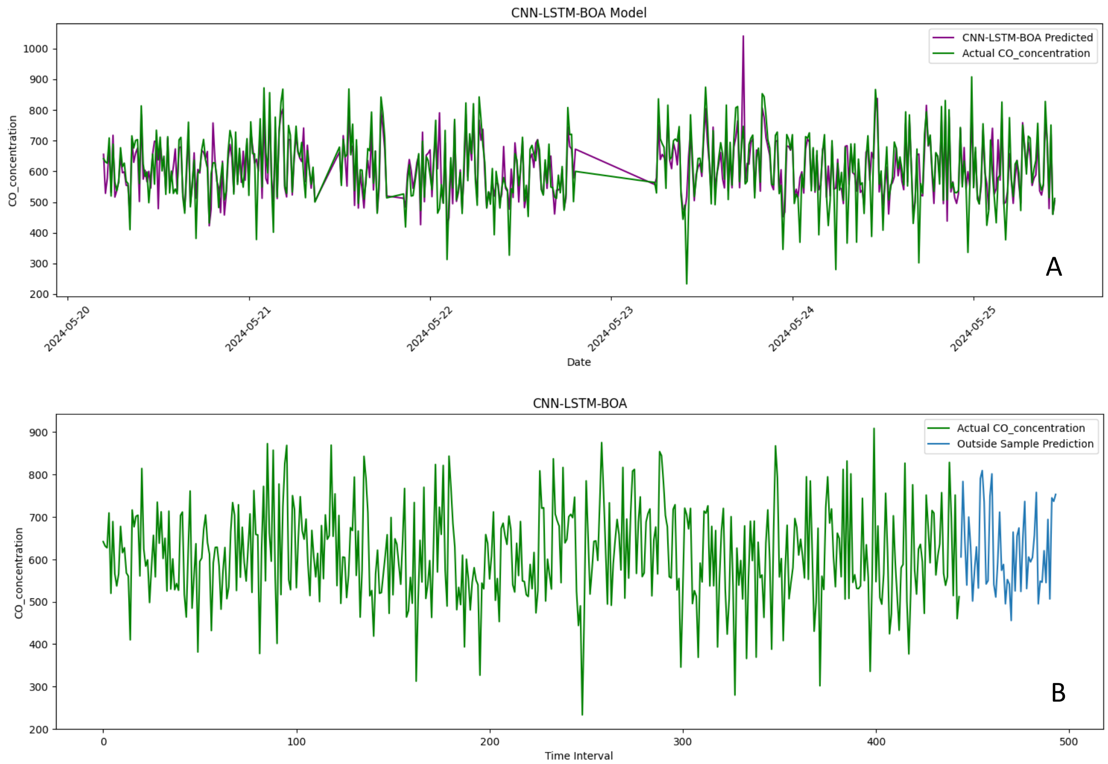


Figure IV.5: CNN-LTSM-BOA Predictions and Actual CO Concentrations

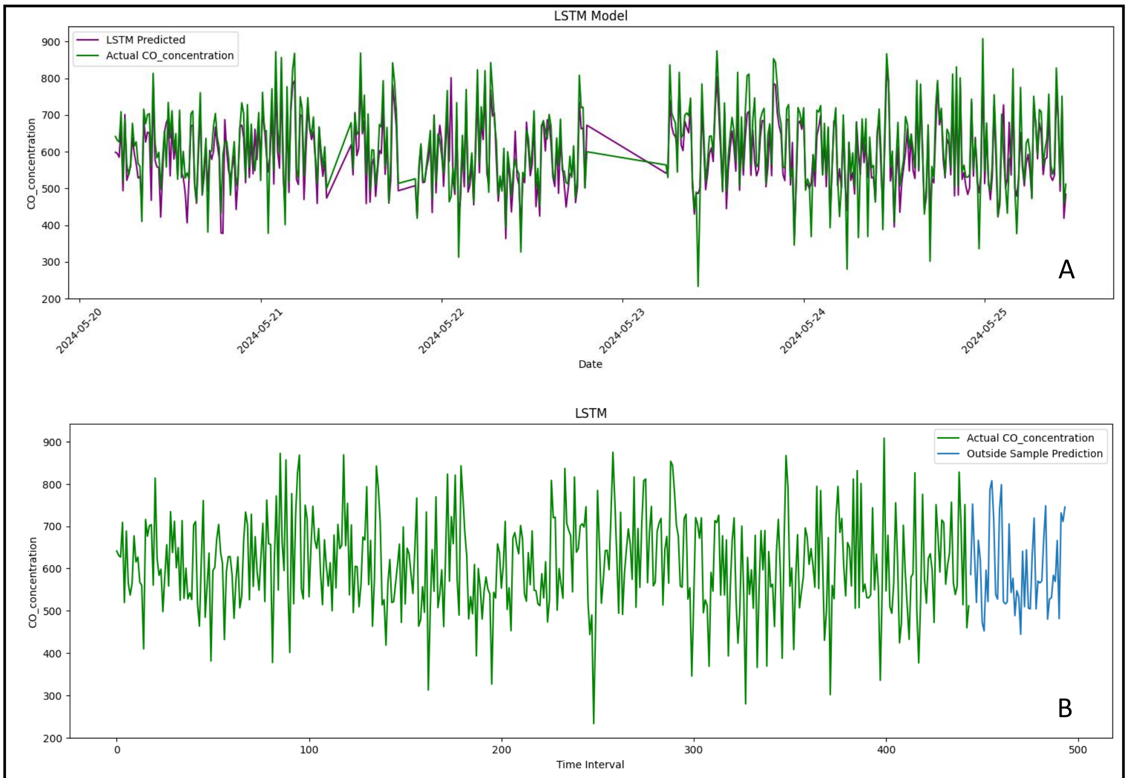


Figure IV.6: LTSM Predictions and Actual CO Concentrations

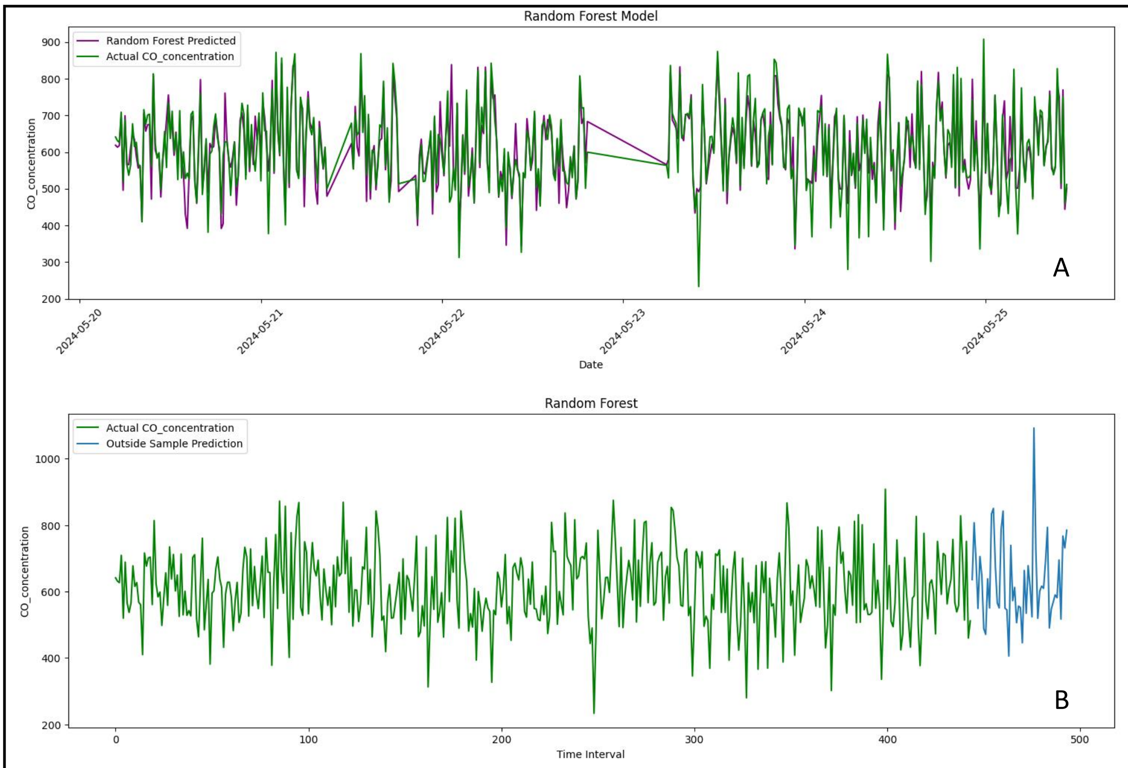


Figure IV.7: Random Forest Predictions and Actual CO Concentrations

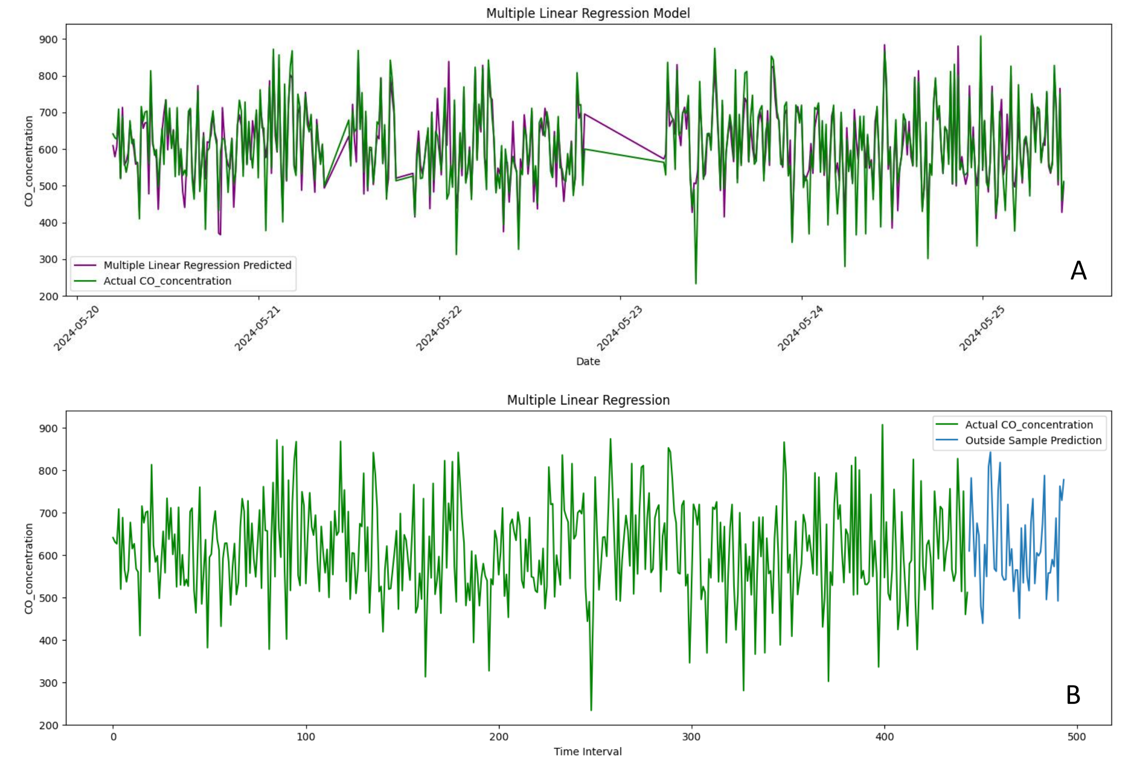


Figure IV.8: Multiple Linear Regression Predictions and Actual CO Concentrations

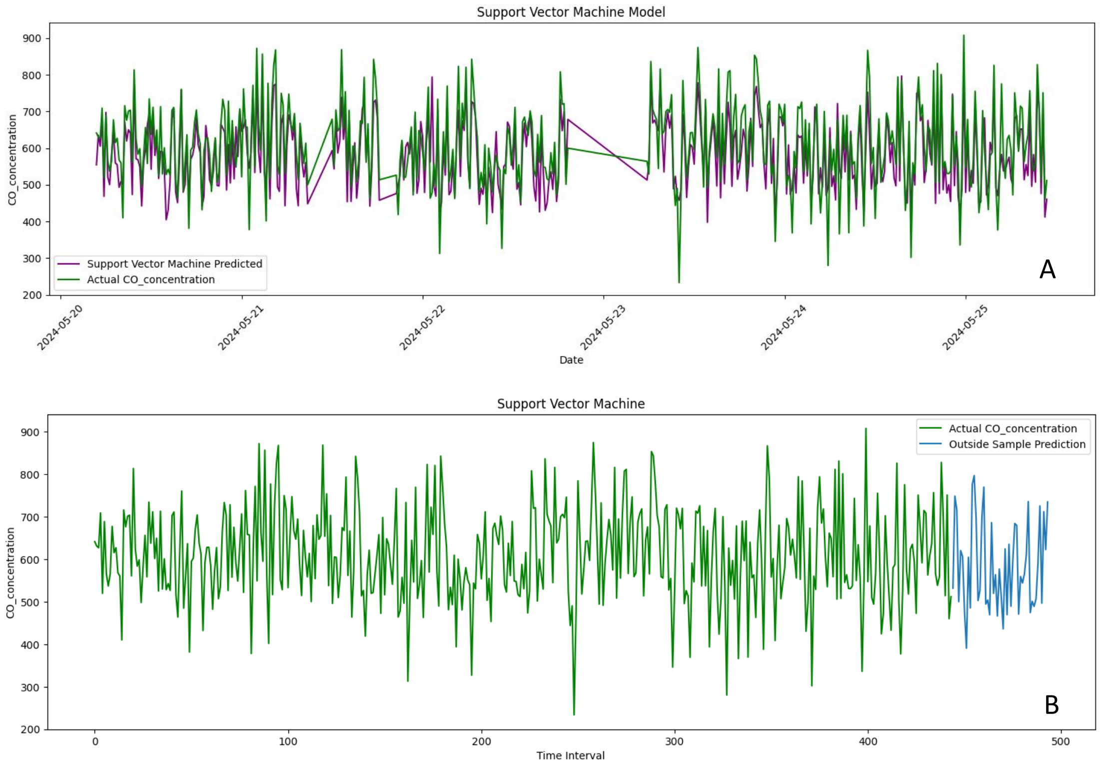


Figure IV.9: Support Vector Machine (SVM) Predictions and Actual CO Concentrations

Performance evaluation:

The **Table IV.2** presents various evaluation metrics for different machine learning models on a regression task. The models evaluated are ANN (Artificial Neural Network), k-NN (k-Nearest Neighbors), CNN (Convolutional Neural Network), CNN-LSTM, CNN-LSTM-BOA, LSTM (Long Short-Term Memory), Random Forest, Multiple Linear Regression, and Support Vector Machine (SVM).

	ANN	**K-NN**	**CNN**	**CNN-LSTM**	**CNN-LSTM-BOA**	**LSTM**	**RANDOM FOREST**	**MULTIPLE LINEAR REGRESSION**	**SUPPORT VECTOR MACHINE**
MAE	***32.282090***	***34.519367***	***39.578394***	***37.743281***	***40.128687***	***39.628481***	***29.299186***	***29.187857***	***50.605759***
MAPE	***0.061992***	***0.063723***	***0.075190***	***0.073478***	***0.076207***	***0.070104***	***0.054408***	***0.055012***	***0.088659***
R-SQUARD	***0.776809***	***0.764142***	***0.677826***	***0.664049***	***0.670178***	***0.774450***	***0.817049***	***0.810182***	***0.686683***
MSLE	***0.010434***	***0.010676***	***0.014814***	***0.015414***	***0.015213***	***0.010170***	***0.008732***	***0.009257***	***0.013364***
RMSE	***53.000045***	***54.483300***	***63.677109***	***65.024418***	***64.428505***	***53.279403***	***47.985045***	***48.877265***	***62.795729***

Table IV.2: Performance evaluation of different models

The evaluation metrics used are:**[25-29]**

- **Mean Absolute Error (MAE) :**

Description: Measures the average absolute difference between actual and predicted values. Lower MAE indicates a better model.

Function: mean_absolute_error(y_true, y_pred)

Mathematical Formula:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | y_i - \hat{y}_i | \qquad \boldsymbol{eq\ IV.1}$$

Where:

- y_i is the actual value.
- $\hat{y}_i$ is the predicted value
- n is the number of samples.

- **Mean Absolute Percentage Error (MAPE):**

Description: Measures the average absolute percentage difference between actual and predicted values. Expressed as a percentage.

Function: mean_absolute_percentage_error(y_true, y_pred)

Mathematical Formula:

$$MAPE = \frac{100}{n}\sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \qquad \boldsymbol{eq\ IV.2}$$

- **R-squared (Coefficient of Determination):**

Description: Measures the proportion of the variance in the dependent variable that is predictable from the independent variables. An R^2 value close to 1 indicates a good model.

Function: r2_score(y_true, y_pred)

Mathematical Formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \grave{y})^2} \qquad \boldsymbol{eq\ IV.3}$$

Where: $\hat{y}_i$ is the predicted value

- **Mean Squared Logarithmic Error (MSLE):**

Description: Measures the mean squared difference between the logarithm of actual and predicted values. Useful when both actual and predicted values span a wide range.

Function: mean_squared_log_error(y_true, y_pred)

Mathematical Formula:

$$MSLE = \frac{1}{n}\sum_{i=1}^{n}(\log(1 + y_i) - \log(1 + \hat{y}_i))^2 \qquad \boldsymbol{eq\ IV.4}$$

- **Root Mean Squared Error (RMSE):**

Description: Measures the square root of the average squared differences between actual and predicted values. Lower RMSE indicates a better model.

Function: np.sqrt(mean_squared_error(y_true, y_pred))

Mathematical Formula:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad \boldsymbol{eq\ IV.5}$$

Table (IV.2) analysis

Lower values of MAE, MAPE, MSLE, and RMSE indicate better model performance, while higher values of R-squared (closer to 1) indicate a better fit between the model's predictions and the actual target values.

Based on the evaluation metrics, the Random Forest model performs the best overall, with the lowest MAE (29.299186), MAPE (0.054408), MSLE (0.008732), and RMSE (47.985045), as well as the highest R-squared value (81.7048%).

The Multiple Linear Regression model also performs well, with the second-lowest MAE (29.187857), MAPE (0.055012), MSLE (0.009257), and RMSE (48.877265), and a high R-squared value (81.0182%).

The Support Vector Machine (SVM) model has the same R-squared value (68.6683%) as the Multiple Linear Regression model, but its other metric values (MAE, MAPE, MSLE, and RMSE) are higher, indicating poorer performance.

The remaining models (ANN, k-NN, CNN, CNN-LSTM, CNN-LSTM-BOA, and LSTM) generally perform worse than the Random Forest and Multiple Linear Regression models, with higher error metric values and lower R-squared values.

It's worth noting that the ANN model has the lowest R-squared value (0.2171%), indicating a very poor fit between its predictions and the actual target values.

While the choice of the best model should consider the trade-offs between different metrics and the specific requirements of the problem, based on the provided evaluation metrics, the Random Forest model appears to be the best-performing model, followed by the Multiple Linear Regression model.

IV.4 Conclusion:

The evaluation of various predictive models for forecasting environmental parameters has demonstrated that the Random Forest model performs the best overall, followed by the Multiple Linear Regression model. These models provide accurate and reliable predictions, which are crucial for effective environmental management and policy-making. The analysis highlights the importance of selecting appropriate models based on performance metrics to achieve precise and robust predictions. This study contributes valuable insights into the strengths and weaknesses of different machine learning models in the context of environmental data analysis.

General Conclusion:

this work successfully addresses the critical challenge of monitoring air quality through the development and implementation of an intelligent air quality monitoring system. By leveraging the synergistic potential of the Internet of Things (IoT) and Artificial Intelligence (AI), the system demonstrates a robust and scalable solution for real-time assessment of various environmental parameters.

The research integrates hardware components, including an Arduino Nano microcontroller, gas sensors (MQ-7, MQ-9, MQ-135), a temperature and humidity sensor (DHT11), a Wi-Fi module (ESP01), and a micro-SD card module. These components collectively enable precise data acquisition. The software aspects encompass programming within the Arduino Integrated Development Environment (IDE), data aggregation and visualization via the ThingSpeak IoT platform, and predictive modeling using Google Colab's AI tools.

The empirical findings underscore the efficacy of the system in providing accurate and real-time data on air pollutants such as carbon monoxide, methane, and ammonia, along with temperature and humidity levels. The predictive capabilities of the AI models, evaluated through metrics like Mean Absolute Error, Mean Absolute Percentage Error, and Root Mean Squared Error, highlight the system's potential in forecasting air quality with considerable accuracy.

This work not only contributes to the field of environmental monitoring but also offers practical implications for public health and environmental management. The integration of IoT and AI technologies provides stakeholders with enhanced tools for proactive and informed decision-making to combat air pollution effectively.

Looking ahead, several avenues for future research and development can further enhance the capabilities and applications of this air quality monitoring system:

1. **Integration with Additional Sensors**: Future iterations of the system could incorporate additional sensors to monitor a wider range of pollutants, such as nitrogen dioxide (NO_2), sulfur dioxide (SO_2), and particulate matter (PM2.5 and PM10). This would provide a more comprehensive overview of air quality.

2. **Advanced AI Models**: Exploring more advanced AI models and machine learning techniques, such as ensemble learning and deep reinforcement learning, could improve the accuracy and robustness of air quality predictions.
3. **Scalability and Deployment**: Scaling up the system for deployment across larger geographical areas could enhance its utility. Implementing a network of interconnected monitoring units could provide granular and widespread data coverage, facilitating better regional air quality management.
4. **Energy Efficiency**: Optimizing the system for energy efficiency, particularly for remote and long-term deployments, would be beneficial. Investigating low-power hardware solutions and energy harvesting methods could support sustainable operation.
5. **Real-Time Public Alerts**: Developing a user-friendly interface for real-time public alerts and integrating with mobile applications could empower communities with timely information about air quality, aiding in public health interventions.
6. **Cross-Disciplinary Collaboration**: Collaborating with environmental scientists, public health experts, and policymakers can help translate the technical capabilities of the system into actionable strategies for pollution control and health risk mitigation.

By addressing these future directions, the intelligent air quality monitoring system can evolve into a more powerful tool, capable of making significant contributions to environmental sustainability and public health.

Bibliographies

1. Joseph, J. *Different Types of Arduino Boards - Quick Comparison on Specification & Features.*; Available from: https://circuitdigest.com/article/different-types-of-arduino-boards.
2. Natheem S, A., *TOP 100 ARDUINO PROJECT FORINNOVATORS*. 2 ed. 2022.
3. Söderby, K. *"Getting Started with Arduino | Arduino Documentation." Arduino Documentation*. Available from: https://docs.arduino.cc/learn/starting-guide/getting-started-arduino/.
4. *"Arduino - Home." Arduino*. Available from: , https://www.arduino.cc/.
5. Svrzić, J., *C Programming for Arduino: Learn how to program and use Arduino boards with a series of engaging examples, illustrating each core concept*. 2013.
6. Teseny, P.A., *Arduino Language Reference.*
7. Kurni, M., M.S. Mohammed, and S.K. G, *A Beginner's Guide to Introduce Artificial Intelligence in Teaching and Learning.* 2023. : Springer Nature Switzerland AG.
8. Serpanos, D., et Marilyn Wolf, *Internet-of-Things (IoT) Systems: Architectures, Algorithms, Methodologies.* 2018: Springer International Publishing.
9. *MQ-7 Gas Sensor Technical Data.* . Available from: http://www.hwsensor.com
10. *MQ-9 Semiconductor Sensor for CO/Combustible Gas Datasheet.*; Available from: http://www.hwsensor.com
11. *MQ-135 Gas Sensor Technical Data.*; Available from: http://www.hwsensor.com
12. *Gas Sensor Pinout*. Available from: https://mavink.com/post/077751673CDA89A150DA28230505C627B8AM4A3FA9/gas-sensor-pinout.
13. Bales, S., *Arduino Measurement Projects for Beginners: Arduino Programming Basics and Get Started Guide*. 2018: Amazon Digital Services LLC- KDP Print US. .
14. *ESP8266EX Datasheet* 2022; Available from: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf
15. DELPORT, R. *INTRODUCING THE INEXPENSIVE ESP-01 WI-FI MODULE*. 19 DECEMBER 2017; Available from: https://behind-the-scenes.net/introducing-the-inexpensive-esp-01-wi-fi-module/.
16. *Micro SD Card Module for Arduino [Product description].* Available from: https://www.adafruit.com/product/254
17. *MB102 Breadboard 3.3V/5V Power Supply [Data sheet]*. 2015; Available from: https://www.handsontec.com/dataspecs/mb102-ps.pdf
18. *Breadboard Connections, Features, Circuit Examples & Datasheet* 2018; Available from: https://components101.com/misc/breadboard-connections-uses-guide.
19. HEMMINGS, M. *What is a Jumper Wire?* . 2018; Available from: https://blog.sparkfuneducation.com/what-is-jumper-wire.
20. *ThingSpeak Internet of Things.*
21. *KoboldAI. (n.d.). KoboldAI-Client/GPU.ipynb at main · KoboldAI/KoboldAI-Client.* . Available from: https://github.com/KoboldAI/KoboldAI-Client/blob/main/colab/GPU.ipynb
22. *Hugging Face on Colab. Google Colab*. Available from: https://github.com/KoboldAI/KoboldAI-Client/blob/main/colab/GPU.ipynb
23. Ainova. *【無料】Google Colaboratoryで使えるコード生成AI「Colab AI」を徹底紹介！料金や日本語での使い方など | Ainova.* . 2023, April 5; Available from: https://generativeinfo365.com/?p=1857.
24. Kataria, A., *AI- and IoT-based hybrid model for air quality prediction in a smart city with network assistance.* . IET Networks, 2022.
25. Andreev, I. *What is predictive analytics?"* 2019; Available from: www.valamis.com/hub/predictive-analytics.

26. *What is Predictive Modeling? Types & Techniques*. Available from: www.qlik.com/us/data-analytics/predictive-analytics. .
27. Hayes, A. *Predictive Analytics: Definition, Model Types, and Uses*. 2021; Available from: www.investopedia.com/terms/p/predictive-analytics.asp. .
28. Saleh, A.Y. *INVESTIGATION OF PERFORMANCE METRICS IN REGRESSION MODELS FOR PREDICTIVE ANALYTICS*. 2018; Available from: www.researchgate.net/publication/326219210_INVESTIGATION_OF_PERFORMANCE_METRICS_IN_REGRESSION_MODELS_FOR_PREDICTIVE_ANALYTICS. .
29. Glen, S. *A Beginner's Guide to Predictive Analytics*. 2022; Available from: www.datacamp.com/blog/predictive-analytics-guide.