



UNIVERSITY of
RWANDA



COLLEGE OF SCIENCE AND TECHNOLOGY

Edge AI Based Respiratory Disease Recognition from Exhaled Breath Signatures

By:

Samson Otieno Ooko

Ref: 220014196

A dissertation submitted in partial fulfilment of the requirements for the degree of

Master of Science in Internet of Things- Wireless Intelligent Sensor Networks

In the College of Science and technology

Supervised by: Dr. Jimmy Nsenga

Co supervisor: Dr. Didacienne Mukanyiligira

Student Declaration

I declare that this Dissertation contains my work except when specifically acknowledged.

Samson Otieno Ooko

220014196

Signature:



Date: **02/11/2021**


Bonafide Certificate

This is to certify that the project entitled “Edge AI Based Respiratory Disease Recognition from Exhaled Breath Signatures” is a record of original work done by Samson Otieno Ooko with registration number 220014196 in partial fulfillment of the requirement for the award of masters of sciences in Internet of Things in College of Science and Technology, University of Rwanda, the Academic year 2020/2021

This work has been submitted under the guidance of Dr. Jimmy Nsenga and Dr. Didacienne Mukanyiligira

Main Supervisor: Dr. Jimmy Nsenga

Co-Supervisor: Dr. Didacienne Mukanyiligira

Signature:


Signature:


The Head of Masters and Training

Dr. James Rwigema

Signature:

Date:

Acknowledgements

I am grateful to the almighty God for blessings upon my life and for leading me this far. I thank him for the good health and resources that made it possible for me to undertake this research.

I would like to acknowledge the invaluable support offered by my supervisors' Dr. Jimmy Nsenga, Dr. Didacienne Mukanyiligira, and Dr. Jean Pierre Munyampundu. I thank you for your dedicated support and guidance throughout the process. I do appreciate your encouragement and willingness to share your experiences. You have installed in me a sense of self-confidence and adequacy to conduct this and future research. I have learned a lot from your supervision and I will be forever grateful.

My sincere appreciation goes to my loving family; to my dear wife Bedina, my sons Dylan, Declan, and Dell, my parents, my brothers, and sisters for your prayers and encouragement during this research. I thank you for your patience, moral support, and understanding.

I also appreciate the moral support of the staff from ACEIoT and my colleagues and friends, specifically Rosemary, Marvin, and Mataifas. Thank you for your encouragement, ideas and for being there. To all my lecturers that saw me through the course. I thank you.

It is impossible to register the names of all those who assisted me, I appreciate you all, for everything glory and honor be to God, I have come this far.

ABSTRACT

Every year, about 4 million people die from respiratory diseases. While early prediction would reduce this mortality rate, till now diagnosis is only done at hospitals involving costly diagnosis resources and scarce healthcare professionals. Ideally, regular noninvasive breath analysis check-ups at home would allow us to anticipate medical consultation. Considering developing country's contexts, existing commercial portable diagnosis kits under proprietary licenses are expensive and require internet connectivity to communicate with the remote server running their cloud prediction analytics. Thanks to recent advances in open source edge AI frameworks, this study presents a design of an offline portable kit locally embedding a tiny Machine Learning (TinyML) trained model to predict respiratory diseases. Evaluated on an open dataset of Chronic Obstructive Pulmonary Disease (COPD), the resulting real-time requirements of our edge AI model is 15.9 Kb of ROM and 1.5 Kb RAM and performs the inference in 1 ms. In addition, the use of synthetic exhaled breath data to train an Edge AI model in cases of low datasets was also evaluated with the model performance giving accuracies similar to that based on actual datasets. Results also show that the accuracy and peak memory for the model are affected by pre-processing, type of sensors, and the number of sensors. In addition to early detection of respiratory diseases, the proposed solution will be of great value in the process of mass collection of exhaled breath data to complement synthetic data and the few breaths that are collected in healthcare facilities. This will enable the training of efficient AI models for respiratory diseases. Last but not least, the research results from this master thesis have been published in 3 IEEE scopus-indexed conferences.

Keywords: Breathomics; Edge AI; Respiratory Diseases; Diagnostic Kit; Synthetic Data;

LIST OF ACRONYMS

AI- Artificial Intelligence

BLE- Bluetooth Low Energy

COPD- Chronic Obstructive Pulmonary Disease

CSV- Comma Separated Values

DCR- Distance to Closest Record

GND- Ground

IMS- Identical Match Score

IoT – Internet of Things

ISR- Interrupt Service Routine

LCD- Liquid Crystal Display

LCD- Liquid Crystal Display

LED- Light Emitting Diode

LED- Light Emitting Diode

ML- Machine Learning

NN- Neural Network

NNDR- Nearest Neighbour Distance Ratio

RAM- Random Access Memory

ROM- Read-Only Memory

VOC- Volatile Organic Compounds

WHO- World Health Organization

LIST OF FIGURES

Fig. 1. Mortality rates due to chronic respiratory diseases.....	3
Fig. 2. Edge AI Vs Cloud AI Architecture.....	4
Fig. 3. Global Internet Penetration.....	5
Fig. 4. Our Edge AI tool stack.....	20
Fig. 5. Machine Learning Process.....	20
Fig. 6. Synthetic Data Generation Steps.....	21
Fig. 7. Dynamic Inference process.....	22
Fig. 8. Agile Software Development Steps.....	23
Fig. 9. High-level System Architecture.....	25
Fig. 10. IoT System Architecture.....	25
Fig. 11. System Block Diagram.....	26
Fig. 12. Arduino Nano BLE Sense pin layout.....	27
Fig. 13. MQ-3 Module Pinout.....	28
Fig. 14. MQ-135 Sensor pin layout.....	28
Fig. 15. MQ-137 Sensor pin layout.....	29
Fig. 16. MQ-138 Sensor.....	30
Fig. 17. 16X2 LCD pinout diagram.....	30
Fig. 18. System flow charts.....	32
Fig. 19. Use case diagram.....	33
Fig. 20. Architecture for Training our Edge AI Model.....	34
Fig. 21. Clock Configuration.....	34
Fig. 22. Board pinout.....	35
Fig. 23. Proteus Simulation Model.....	36
Fig. 24. Embedded system prototype.....	38
Fig. 25. Data collection device set up.....	39
Fig. 26. Raw data – COPD.....	40
Fig. 27. Confusion matrix for the model on the validation dataset.....	41

Fig. 28. COPD infected Sample Simulation Output on both the LCD and the serial monitor....	42
Fig. 29. Inference results on real board.....	43
Fig. 30. Browser prediction output.....	44
Fig. 31. Sample prediction output from Node.js.....	45
Fig. 32. Similarity based and privacy tests.....	47
Fig. 33. Holdout Method.	48
Fig. 34. Raw data sample.....	48
Fig. 35. Generated synthetic data features.....	49
Fig. 36. Synthetic data model confusion matrix.....	49
Fig. 37. Real Time embedded device resource usage.....	50
Fig. 38. Effect of feature reduction on accuracy and memory.....	51
Fig. 39. On-device performance as the number of sensors is increased.....	52
Fig. 40. Individual Sensor performance.....	53
Fig. 41. Sensor combination performance.....	53
Fig. 42. open dataset model confusion matrix.....	54
Fig. 43. Synthetic data model confusion matrix.....	54
Fig. 44. Synthetic data model test result.....	56
Fig. 45. A plot of data size vs accuracy of the model.....	57

TABLE OF CONTENTS

Student Declaration	ii
Bonafide Certificate	iii
Acknowledgements	iv
ABSTRACT	v
LIST OF ACRONYMS	vi
LIST OF FIGURES	vii
CHAPTER 1	1
INTRODUCTION	1
1.0 Introduction	1
1.1 Problem Statement	4
1.2 Aims and Objectives	6
1.2.1 Aims	6
1.2.2 Specific Objectives	6
1.3 Hypothesis	6
1.4 Research Questions	7
1.5 Study Scope	7
1.6 Significance of the Study	7
1.7 Organization of the Document	8
CHAPTER 2	9
LITERATURE REVIEW	9
2.1 Breathomics: the science of exhaled breaths	9
2.2 Solutions for predicting respiratory diseases from exhaled breath	10
2.3 Transforming exhaled breaths into data	14
2.4 Breath Analytics	14

2.5 Using ML to predict respiratory diseases from exhaled breath datasets	15
2.6 Use of synthetic data generation for ML	16
2.7 Conclusion	16
CHAPTER 3	17
RESEARCH METHODOLOGY	17
3.1 Research Process	17
3.2 Edge AI Process	19
3.3 Machine Learning Process	20
3.4 Synthetic Data Generation	21
3.5 Anomaly detection	22
3.6 Use of Dynamic Inference	22
3.7 System Design Method	23
CHAPTER 4	24
SYSTEM ANALYSIS AND DESIGN	24
4.1 System Architecture	24
4.1.1 High-level System Architecture	24
4.1.2 IoT System Architecture	25
4.2 Embedded System-Level Design	26
4.2.1 System Block Diagram	26
4.2.2 Hardware Components	26
4.2.3 System PDL	31
4.2.4 System Flow Charts	31
4.3 Edge AI model training architecture	32
4.4 System Simulation	33
4.4 System Simulation	34

4.4 Embedded Device Set-Up	36
CHAPTER 6	39
SYSTEM RESULTS AND ANALYSIS	39
5.1 Edge AI model	39
5.1.1 Open datasets	39
5.1.2 Training Steps	40
5.1.3 Training parameters and output	41
5.1.4 Inference on a virtual embedded board	41
5.1.5 Inference on the real embedded board	42
5.1.6 Inference on local machines	44
5.2 Experimentation on use of synthetic data	45
5.2.1 Input: Raw COPD Dataset	45
5.2.2 Synthetic Data Generation	45
5.2.3 Synthetic Data Quality Measurement	46
5.2.4 Embedded ML model generation	47
5.2.5 Synthetic data generation results	49
CHAPTER 6	50
DISCUSSIONS	50
6.1 Real-time embedded processor specifications	50
6.2 Impact of Preprocessing	51
6.3 Importance of different sensors in inference accuracy	52
6.4 Applicability of Synthetic Data	54
6.5 Comparison to related models	55
6.6 Synthetic data generation results	56
6.7 Evaluation of synthetic data Model Performance on test data	56

6.8 Effect of data size on model Accuracy	57
CONCLUSION	58
REFERENCES	59
APPENDICES	64
Appendix 1: Paper Acceptance notifications	64
Appendix 2: Publication Peer Reviews	65
Appendix 3: Thread on inference simulation attracting many readers	66

CHAPTER 1

INTRODUCTION

1.0 Introduction

Respiratory diseases affect passages of the air, including the lungs, bronchi, bronchioles, and nasal passages. Conditions considered respiratory-related include not only acute respiratory infections, such as common cold, sinusitis, pharyngitis, epiglottitis, and tracheobronchitis but also chronic respiratory diseases, such as lung cancer, asthma, and chronic obstructive pulmonary disease [1].

According to WHO reports, [2] Respiratory diseases are one of the major causes of disabilities and deaths across the globe with over four million people dying prematurely from such disease. Chronic Obstructive Pulmonary Disease (COPD) is the third leading cause of death in the globe with about sixty-five million people suffering from the disease and over three million dying from it each year [2]. Asthma is one of the most common chronic childhood diseases affecting fourteen percent of children globally with over three hundred and thirty-four million people suffering from the disease globally [2]. Pneumonia is not only the leading cause of death in children under the age of five but also leads to the death of millions of people yearly. Some respiratory diseases are also highly infectious, with the most lethal being tuberculosis (TB) that affects over ten million and kills over one million people each year. Lung cancer is considered the deadliest form of cancer-killing almost two million people annually [2].

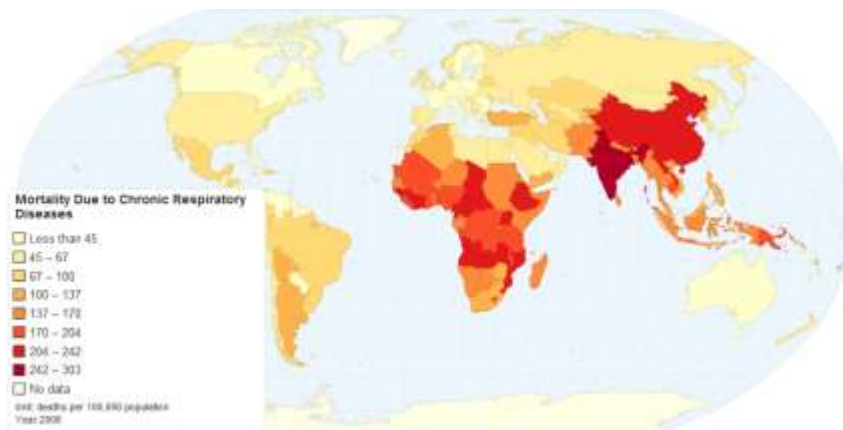


Fig. 1: Mortality rates due to chronic respiratory diseases [3]

Fig. 1 shows mortality rates due to chronic respiratory diseases globally. The high rate of mortality results from late diagnosis occurring when a patient starts to experience symptoms and go to hospitals. Diagnosis in healthcare facilities requires expensive resources such as equipment and healthcare professionals to envision regular preventive check-ups for all populations. Affordable, free-to-use, noninvasive early prediction solutions for home use would anticipate on-time medical consultation and therefore treatment, and at the same time increase the datasets of biomarkers required to develop improved respiratory disease prediction analytics.

The emerging concept of Internet of Things (IoT) is increasingly being used in the healthcare sector and provides capabilities that can be exploited in respiratory disease management from early detection to collection of medical data and monitoring. The use of the Internet of things offers many benefits by enhancing efficiency and effectiveness in remote health monitoring as compared to the traditional methods of health monitoring. IoT-enabled technologies allow for smaller, cheaper, and portable devices enabling patients to use them at any location, at any time for both monitoring and diagnosis of health conditions [4]. The use of IoT also allows for real-time monitoring of patients with critical conditions over the internet enabling immediate medical actions and remedies when the need arises. Moreover, IoT devices enhance the collection of a variety of voluminous health data at high velocity [4]. Further analytics can be performed on the collected data to improve diagnosis and decision-making in the health sector.

With the integration of IoT and Artificial Intelligence, methods for example knowledge description, machine learning, deep learning, and expert systems, are used in the design, development, and implementation of medical applications that enhance precision medicine [5]. The most common techniques of AI used in healthcare applications are knowledge and expert-based systems. In cases where it's not possible to design expert systems due to inadequate knowledge, the collected data relating to a clinical case can be analyzed using machine learning techniques to systematically describe a clinical knowledge that can characterize the clinical condition of a disease or patient [5]. The techniques can also be applied in predicting respiratory diseases using the data collected by the IoT sensing devices.

Some integration of IoT and AI in predicting respiratory diseases include the use of computed tomography analysis, x-ray image analysis, forced oscillation tests, lung sound analysis, and exhaled breath analysis [6]. Particularly today, the advances in chemical-based sensor technologies

enable the development of cheaper non-invasive embedded systems commonly known as e-nose [7] that sense volatile organic compounds (VOC) from collected breath profiles. However, breath AI analytics for respiratory disease prediction is still in its infancy due to the lack of enough datasets of breath prints [8]. Open datasets in healthcare are likely to continue being a challenge given the strict privacy requirements limiting open sharing and also the high costs involved in collecting, curating, and maintaining high-quality data.

The availability of enough and quality datasets is an essential prerequisite in developing more accurate predictive machine learning (ML) models. Other conventional issues with health data sets include missing data, inconsistencies with datasets, and replicated data [9]. In the area of exhaled breath data, available datasets are based on traditional diagnostic methods. There are limited datasets collected from the emerging IoT sensing technologies which are also mostly closed due to privacy concerns hindering the development of better ML models for the prediction of respiratory diseases. The use of synthetic data is a possible solution to the problem of datasets that can enhance the rapid development and validation of better ML models for predicting respiratory diseases [10]. Good synthetic data captures the distributions and dependencies of a real dataset which at the same time preserves privacies. Open source tools can be used to generate synthetic data with precision and accuracy.

Existing studies [11], [12] and commercial solutions [13] integrating IoT and ML for portable non-invasive breath analysis use an overall system architecture design centered around the cloud, meaning that the breath biomarkers collected by IoT sensors need to be sent to the cloud where the disease prediction analytics are executed. However, considering the African context, especially in rural areas, such an online design solution is not viable due mainly to limited and costly internet connectivity and also energy consumption of wireless communication technologies. Therefore, an offline breath analysis solution is necessary. By moving the AI analytics at the edge, a technique known as edge AI [14], the design will lead to reduced costs, enhanced privacy, real-time diagnosis, and portability to any location without the need for connectivity concerns. Fig 2. Illustrates a comparison between a cloud centric and edge centric AI models. With the Edge AI the ML intelligence is moved to the data source unlike in the cloud centric architecture where the data is transported to the cloud to enjoy from ML intelligence.

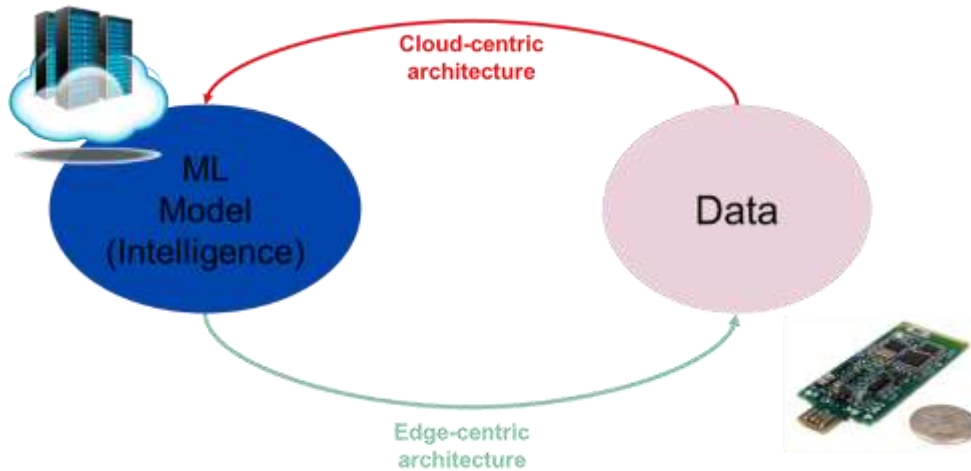


Fig. 2: Edge AI Vs Cloud AI Architecture

It is in this context that this thesis presents a prototype design of an edge AI portable embedded respiratory disease prediction kit as a way to (1) enable offline free to use and regular checkups in home settings and (2) collect large curated datasets to enable the development of analytics achieving clinical-grade accuracy. We base our design on open-source prototyping toolsets and evaluate the edge AI inference using an open dataset of COPD as training data. Our design is scalable to other diseases as long as corresponding datasets are available. This study also explored the use of open-source tools to generate synthetic data for the training of an Edge AI model to predict COPD. The results were compared against a model based on real datasets as proof of the capabilities of using synthetic data for ML as a solution to the dataset problem.

This thesis fits in the context of applying IoT and AI for detecting respiratory diseases from exhaled breath. The implementation and realization of IoT for detection and monitoring of respiratory diseases is needed in Rwanda given the risk of getting more severe symptoms of respiratory diseases resulting from household air pollution that is now also a major concern in the country. The proposed system will also go a long way in helping the health authorities collect breath data that can be used for further research and planning.

1.1 Problem Statement

With the population size, Africa witnesses a relatively high rate of respiratory mortality and morbidity. The most common and highly infectious respiratory diseases in Africa include pneumonia and tuberculosis. High rates of infection by non-communicable respiratory diseases

such as asthma and COPD have also been reported [15]. According to an article in the New Times, It was reported that over 3 million Rwandan nationals suffer from respiratory problems annually. This can be translated to 1 out of 4 persons suffering from respiratory-related diseases yearly [16]. With emerging highly contagious and deadly respiratory diseases like COVID-19, there is a need to have a regular and cheap diagnosis to avoid large spreading. Currently, diagnosis is only carried in healthcare facilities in case the infected patient goes for consultation, most likely when his/her symptoms become severe. Quick, cheap self-diagnostic tools that individuals can use at home are thus becoming important in our daily life.

Even though there have been attempts to design devices that can be used for self-diagnostics, they are still faced with several challenges. To begin with, collecting biological information necessary for diagnosis involves using expensive equipment in the laboratories and consultation of few and busy medical experts, making people waste both valuable time and resources. Secondly, existing commercial devices are either not tailored for home use or expensive for the African population (For example trio-smart [16] costs \$289 per unit). In addition, the solutions are centered around cloud analytics, an approach that is not appropriate for implementation in Africa due to connectivity, power, and privacy challenges. It is reported in [17] that the penetration the internet in Africa is the least in the world with 170 million users translating to 18% of the population which is significantly lower than the global average of 30 percent.

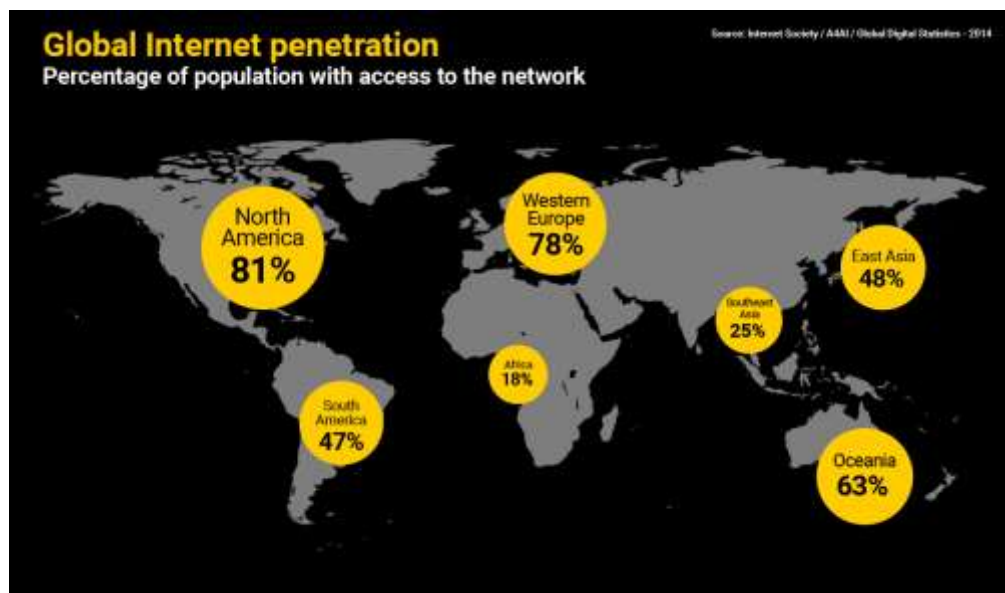


Fig. 3: Global Internet Penetration [17]

Fig. 3 gives a global view of internet penetration with African having the least penetration at 18 percent. There is therefore a need for a solution that will not only be portable but also affordable to the African population with the capability to diagnose respiratory diseases from home without relying on the cloud.

1.2 Aims and Objectives

1.2.1 Aims

The aim of this research study is to design and prototype a smart portable respiratory disease diagnostic device for day-to-day use considering the African context challenges such as limited budget, connectivity and energy supply. This aim is to be achieved by leveraging the latest advances in open source IoT, edge computing and AI technologies running on edge devices known as Tiny ML.

1.2.2 Specific Objectives

The following specific objectives have been defined:

- I. To understand how respiratory diseases can be detected from exhaled breath,
- II. To identify IoT and embedded technologies that are used in sensing exhaled breath signatures and transmitting them from edge devices to the cloud
- III. To identify open datasets for training ML model for respiratory disease detection and also explore ways to synthetically generate datasets while preserving original data patterns
- IV. To identify open-source AI platforms that can be used to train ML targeted to be executed on limited real-time resources devices.
- V. To design, simulate and prototype a smart non-invasive embedded personalized diagnostic kit for respiratory diseases.
- VI. Publish the results of our research to high quality conferences

1.3 Hypothesis

Our hypotheses were as follows; 1) There are existing open data sets that can be used to train an AI model to predict respiratory diseases from exhaled breath. 2) Open source technologies can be exploited to train an Artificial Intelligence Model and optimize it for the detection of respiratory diseases on an embedded device from exhaled breath.

1.4 Research Questions

The study was guided by the following questions;

- I. How can respiratory diseases be detected from exhaled breath?
- II. How can IoT be integrated with AI to detect respiratory diseases from exhaled breath?
- III. How can AI models be trained to be deployed on embedded devices for inference?
- IV. How to detect anomalies in exhaled breath data before inferencing? The study targets to infer only human exhaled breaths
- V. How can an embedded system with local AI be simulated without a HW board?
- VI. How to trade-off the number of exhaled breath sensors with respect to inference accuracy and costs?

1.5 Study Scope

Due to limited resources, and availability of datasets for the training of an AI model, this study was evaluated on the prediction of COPD based on open datasets and synthetically generated data.

1.6 Significance of the Study

The system will be of benefit to the population at large by helping them to be aware of their respiratory health status, and in case of infections be able to seek medical attention as early as possible. Such actions will lead to healthier and more productive citizens leading to economic growth. Hospitalization cases resulting from respiratory diseases will also be greatly reduced leading to fewer strains in the health sector.

Health care professionals will spend their time on complex issues, therefore improving the care of the most in need.

The solution will be of great value in the process of mass collection of exhaled breath data, complementing the few breath data that are collected in healthcare facilities. This will enable the training of efficient AI models for respiratory diseases.

1.7 Organization of the Document

The rest of this document is organized as follows: The next chapter gives a review of related literature; Chapter 3 describes the methodology used in the study, the research process is outlined, the ML process, the edge AI pipeline, synthetic data generation and the system design methodology and materials also presented; Chapter 4 presents the system design and analysis, the system architecture, the system-level design, the simulation model, and embedded system layout are presented; Chapter 5 System Results and analysis, shows an evaluation in the prediction of COPD by outlining the model training, synthetic data model, the results obtained from both the cloud and the embedded simulated and real system; Chapter 6 Discussion, presents the analysis of trade-offs in different embedded system parameters and also presents a comparison of different models; A conclusion the document by presenting the recommendations drawn from this study.

CHAPTER 2

LITERATURE REVIEW

In this section a review of related literature is presented showing the basis for the Integration IoT, ML, and Edge Computing to Predict Respiratory Diseases. A state of the art on the use of exhaled breath data to predict respiratory diseases is given by; firstly, presenting commercial smart breath devices and related cloud services for respiratory disease detection, secondly, prototyping works that show the system architecture of the exhaled breath sensor, last but not least, a quick state of the art of remote analytics for breath data. The use of ML in predicting respiratory diseases from exhaled breath is also highlighted and finally, the use of synthetic data for ML is described.

2.1 Breathomics: the science of exhaled breaths

Breathomics can broadly be referred to as the metabolic study of exhaled breath. It mainly focuses on the analysis of the volatile organic compounds (VOCs) available in human exhaled breath [18]. The amount of VOCs varies with the health status of an individual, thus breathomics can be used in the noninvasive diagnosis of a range of diseases. This is achieved by either analyzing the quantities of specific disease biomarkers or finding patterns relating to abnormal metabolic processes [18]. The analysis of exhaled breath also has the potential of providing the status information on different metabolic activities taking place in the respiratory tract through the systemic circulation.

A human breath database [19] is proposed to make available breath VOCs, biomedical data, and related references through manually correlating information and automatically extracting biomedical data from different sources. From [19] and [20] some of the identified common biomarkers for respiratory diseases include carbon monoxide, nitrogen oxides, and hydrogen peroxides.

Given the fact that exhaled breath VOCs originate from the respiratory tract and are passed through the lungs during air circulations, they can be thus effective in the prediction and diagnosis of respiratory diseases. VOCs found in exhaled breath can be used to differentiate patients suffering from respiratory disease and healthy control subjects and thus predict those with the disease by analyzing various disease biomarkers.

2.2 Solutions for predicting respiratory diseases from exhaled breath

Commercial solutions that apply smart breath collection are coming up. First, the Spiro Nose [13] integrates electronic nose technology and spirometry for the diagnosis of inflammatory disease and cancer. The device measures exhaled breath in real-time and transmits data securely to the online breath base platform for automated analysis. Second, Breath Tracker Analyzer [21] is used to measure the levels of methane, hydrogen, and carbon dioxide in exhaled breath. The data is used by medical professionals in conjunction with other clinical factors known to them for diagnosis.

Prototype solutions have also been proposed by different authors and innovators. To begin with, in [22] authors present AQTech which is an embedded system that can detect the total volatile organic compounds (VOCs) and carbon dioxide concentration from exhaled breath and also uses a pulse oximeter to collect the levels of oxygen in the blood. Second, Electronic noses (e-noses) are reviewed in [23]. They are based on a variety of sensor arrays with the capability of responding to VOCs and other odorant molecules. E-nose uses pattern recognition to distinguish different VOCs spectrum unlike the traditional methods of mass spectrometry and gas chromatography. In the survey [23] several e-nose related studies show that exhaled breath analysis and profiling can be used in the diagnosis of both respiratory and systemic diseases. IAQD proposed in [24] is an Internet of Things device that is aimed at detecting respiratory diseases by measuring the body temperature using an infrared temperature sensor and detecting volatile organic compounds and carbon dioxide concentration from exhaled breath using a gas sensor.

Other recent studies on disease detection from Exhaled breath using IoT include; Online breath analysis using metal oxide semiconductor sensors (electronic nose) for diagnosis of lung cancer [25], Development of a Compact, IoT-Enabled Electronic Nose for Breath Analysis [11], Microelectronic Gas sensors for Non-invasive Analysis of Exhaled Gases [20], Detection Of Ketosis Using Non-Invasive Method [26]. Table 1 gives a summary of the most recent studies in the detection of diseases from exhaled breath.

Table 1: Disease detection from exhaled breath

Recent studies on disease detection from Exhaled breath using IoT					
Study	Sensors	Targeted Gases	Methodology	Database	Disease
Electronic Nose To Detect Patients with COPD From Exhaled Breath [23]	TGS 826, SP-15A, TGS 921, TGS SP-53, TGS 821, TGS 729, TGS 822	Ammonia, Propane, butane, Hydrogen, Hydrocarbons, Organic solvent agents	Samples were taken from healthy and diagnosed patients	Local database used	COPD
Online breath analysis using metal oxide semiconductor sensors (electronic nose) for diagnosis of lung cancer- [25]	Self-designed sensors	Alkanes Toxic gases (CO, NH ₃ .H ₂ S, NO, CH ₄) OH containing compounds (VOCs, H ₂)	Data from volunteers used as a training set to model prediction	Not specified	Lung Cancer
Electronic nose dataset for COPD detection from smokers and healthy people through exhaled breath analysis [27]	MQ-3, SP-3, TGS 822, TGS 813, TGS 800, MQ-138, MQ-137, and MQ-135		Data collected from identified groups based on 3 categories	Labview V14	COPD

Development of a Compact, IoT-Enabled Electronic Nose for Breath Analysis [11]	CCS811, SGP30, BME680 iAQ-Core ZMOD4410 MiCS-6814 TGS-8100 TGS-2620 AS-MLV-P2	NH3, TVOC, CO, H2, ethanol	The system was tested by taking breath samples before and after healthy volunteers took peppermint capsule	Blink	Used to prove the sensors are applicable in breath analysis
Sensor System Development for Bronchitis Detection from Exhaled Breath-[12]	tgs 2600 SY-HS220 LM35	CO content, humidity, and temperature	Tested with two patients	Local Database	Bronchitis
Microelectronic Gas sensors for Non-invasive Analysis of Exhaled Gases [20]	Chinese gas sensors and customized sensor	NH3, NOx, H2S, acetone, ethanol, and H2O2	Testing of customized sensor	Local	Diabetes, asthma, halitosis, renal disease, and lung cancer
Detection Of Ketosis Using Non-Invasive Method [26]	TGS 822, DHT 11	Acetone	Samples from volunteers before and after fasting	Local	Obesity

Microcontroller Implementation of Support Vector Machine for Detecting Blood Glucose Levels Using Breath Volatile Organic Compounds [28]	commercially obtained VOC sensors	Acetone Methyl, Nitrate Ethanol, Methanol	From each footprint 9 tests were used to train the support vector machine		Blood glucose
A Low-Cost Breath Analyzer Module in Domiciliary Non-Invasive Mechanical Ventilation for Remote COPD Patient Monitoring [29]	Infrared CO ₂ O ₂ sensor: SHT75 A dual (NOX-CO) sensor MiCS-4514 A VOC sensor AS-MLV-P2,	CO ₂ , O ₂ , CO, VOC	Testing of both healthy and unhealthy patients.	OMNIACARE	COPD
Non-invasive Measurement of Blood Glucose by Breath Analysis [30]	MQ138, MQ6, MQ3	Acetone, Alcohol, Propane	Data collected from patients and learning done	Excel	Blood Glucose

These solutions recommend the use of gas sensors to detect disease biomarkers from exhaled breath with data being processed on local computers or cloud platforms. Such solutions have limitations because, (1) They depend on medical professionals for further examination, (2) some are based on local servers and thus not scalable, (3) the use of cloud-based solutions is dependent on connectivity and increases security risks. This, therefore, calls for ML models that can enable detection on the embedded devices. To do so enough datasets will be needed for the targeted diseases and therefore the need to use synthetic data.

2.3 Transforming exhaled breaths into data

The first step towards the digitization of breath is sample collection. The collection of breath samples can be done by the use of polymeric bags, canisters, or sorbent tubes [31]. The use of disposable bags is recommended and where not possible nitrogen can be used as a cleaning agent. Breath samples can also be directly sampled into the analytical hardware. In [32], breath samples are collected in a custom-made tube in which the sensors are also placed. The levels of CO₂ and TVOCs are measured in parts per million (ppm). In [33] commercial Tedlar bags are used to collect breath samples through a one-way valve with three samples collected per patient. The bags are cleaned with nitrogen flows before each sample collection.

Different sensing technologies have been applied in recent studies. In [22], a CSS811 air quality sensor is used to detect the total volatile organic compounds (TVOCs) and carbon dioxide (CO₂) concentration from exhaled breath. A study in [7] uses the enose which is made of seven metal oxide semiconductor sensors that are present in duplicates both inside the sensing unit, to measure VOCs in exhaled breath, and outside the device to measure VOCs in air. In [33], a sensor array of six chemical gas sensors is used to measure VOC levels and operate at their optimal temperature. In [27] an array of 8 gas sensors, SP-3, MQ-3, TGS 822, MQ-138, MQ-137, TGS 813, TGS 800, and MQ-135, are used to detect exhaled breath measurements. In [25] analog semiconductor commercial gas sensors (MQ138, MQ6, and MQ3) were used in detecting VOCs from exhaled breath. In [11] MiCS-6814 Dual Sensor, TGS-8100 and TGS-2620 were used with a TGS 2600 used in [12] These studies prove the capabilities of commercial gas sensors in detection breath VOCs. We conclude that gas sensor technologies are mature and can be relied on to digitize exhaled breaths.

2.4 Breath Analytics

Commercial services and databases have been created by medical companies. Some of which include the Breath Base Solution [34] which provides training for reliable and reproducible exhaled breath measurements and immediate analyses. Breath Biopsy Services [35] offers biomarker discovery and validation solutions by combining mass-spectrometry and breath analysis expertise. These and other commercial solutions are tailored to specific services and are also expensive for our use case.

Available open-source services include OpenChrom [36], open-source software for spectrometry, chromatography, and spectroscopy. OpenChrom is a vendor-independent software in which data from different systems can be imported and analyzed. BALSAM [37] is a web platform that simplifies and automates the process of breath analysis, offering features for peak detection, preprocessing, visualization, feature extraction, and pattern discovery. The use of such solutions may be exploited but not applicable to our solution given the fact that they are cloud-based making it difficult to implement in the African context where connectivity is a challenge.

2.5 Using ML to predict respiratory diseases from exhaled breath datasets

Different ML techniques have been applied in predicting respiratory diseases from exhaled breath datasets as noted in [11]. Data available for ML training determine the performance of the resulting analytics. The use of open datasets such as in [27] can be explored to design better algorithms with better performance metrics.

In [23], authors studied Principal Component Analysis (PCA), a non-parametric statistical technique primarily used for dimensionality reduction in ML. In [25] the dataset was first taken through the pre-processing phase and was then split into 70% training and 30% test sets, classification models were trained using the training data; while the test data was used for the estimation of prediction possibility using logistic regression. The latter was found to be an adequate data-processing approach. Furthermore, two linear regressions are used [20] to analyse exhaled breath data.

In a review [8] it is noted that some of the popular ML models for respiratory disease prediction are cloud-based and include support vector machines (SVM), random forest, and artificial neural networks (ANN). In addition, it is observed that deep learning models such as convolution neural network (CNN) and recurrent neural network (RNN) are becoming more popular lately due to the availability of large labelled datasets, efficient training algorithms, and advances in parallel processing.

When considering bandwidth-constrained or privacy-preserving scenarios, sending collected data from edge sensors to the cloud may respectively not be possible or allowed. This calls for moving ML inference to the edge as reported in [38] which in addition decreases latency for inferencing. Edge AI is still a quite new research domain with a lot of open challenges.

2.6 Use of synthetic data generation for ML

In [10] different methods are proposed for generating synthetic data. First, data perturbation is done by the addition of noise to the original dataset. Second, generative models can be applied to the data to capture correct distributions and relationships either by hand-coding using expert knowledge or inferring from real data by using Bayesian networks or neural networks. Generative Adversarial networks are also used for generating synthetic image data. The use of synthetic data techniques like SMOTE is used in dealing with unbalanced or small datasets and generate data points to supplement existing data [23]. Different open-source tools such as Mostly AI [39] are now available online that can be explored for faster generation of synthetic data in different fields of study.

2.7 Conclusion

The literature review supports the capability of diagnosis of respiratory diseases from exhaled breath and the need for the study. The commercial solutions are expensive and do not use open source technologies. Reviewed prototypes are also mainly based on local databases with some not being portable for home use. In addition, the existing solutions depend on cloud analytics, a technique that is not possible to use in the African context due to poor access and cost of connectivity in many regions. Moreover, Africa lacks datasets of exhaled breaths. The only way to build African exhaled breath big data is to bring such kinds of devices to homes at a cheap price and also further explore the viability of synthesizing exhaled breath data.

In this thesis, we particularly focused on setting up a tool stack for rapid prototyping of edge AI applications and validated the inference accuracy on an open dataset and synthetically enhanced datasets for predicting COPD from exhaled breath.

CHAPTER 3

RESEARCH METHODOLOGY

In this chapter the methodology, tools and process used to conduct the study are outlined. First presented are the steps undertaken to complete the study, this is followed by the AI model creation and synthetic data generation methods and last but not least the methods used in the prototype design and development presented.

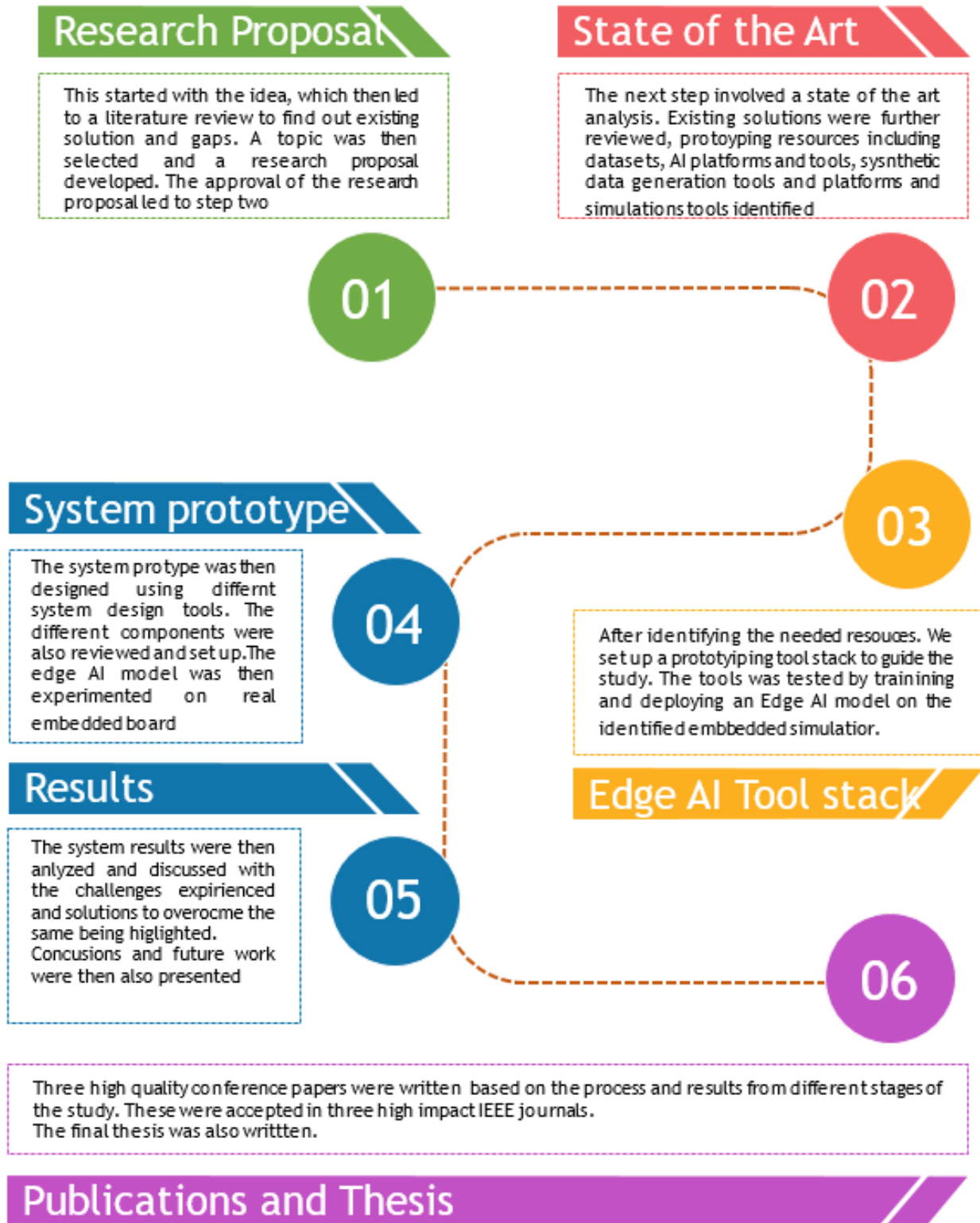
3.1 Research Process

The research process began with an idea that prompted further interrogation by undertaking of a comprehensive literature review. This led to the topic of the study that was formulated based on the identified gaps from literature and existing solutions. The research proposal was then prepared and presented for approval. On approval the steps that followed included;

- (1) State the art analysis
- (2) Identifying prototyping resources (dataset, edge AI platform, synthetic data generator, virtual embedded simulator)
- (3) Setting up a prototyping tool stack
- (4) Training & Deploying edge AI model on Proteus
- (5) System design and experimentation on real embedded board
- (6) Analysis of results & challenges
- 7) Thesis preparation and Publication of results.

These process can be summarized as shown below.

Summary of the research process



3.2 Edge AI Process

As for any ML process, the edge AI process presented in Fig. 4 also requires a dataset for training. We assumed that a dataset had been either collected using the same types of exhaled breath sensors as discussed early or synthetically generated. The next step was to train the dataset with an ML framework that can produce a model optimized for embedded processors known as the TinyML model. The resulting model, was packaged in the form of a software library, and then integrated into our overall application and compiled to the targeted processor architecture with the help of an integrated development environment (IDE). The resulting embedded executable was to be deployed on either an embedded simulator or the targeted board itself. In the simulation context, the performance of edge AI inference was validated by reading test data from a file and comparing the inference accuracy results with the one obtained by ML inference on the cloud using the training platform.

Our tool stacks to implement the above process flow was composed of 3 development environments:

- Edge Impulse: an open-source platform that can be used for the development of ML models for edge devices. It allows direct acquisition of data from the sensing device or uploading of collected data in various formats. Depending on the data different processing blocks are available for training. Different ML techniques may also be applied and in addition to the expert mode, custom processing blocks may be included.
- STM32CubeIDE: a C/C++ IDE that allows for configuration of peripherals, code generation, and compilation of codes, with debug tools for STM32 microprocessors and microcontrollers. It's based on GCC toolchain and Eclipse/CDT framework for development and GDB for debugging. Existing plugins can be integrated to complete the features of Eclipse IDE. It also helps save development and installation time through an all-in-one tool experience by integrating STM32CubeMX project creation functionalities and STM32 configuration.
- Arduino IDE: This is an open software that provides an easy way of writing codes and uploading the same to a wide variety of Arduino development boards. It can run on diffwrt

operating systems including Windows, Linux and Mac OS. Java has been used to write the environment that is also based on processing and other (open source) software.

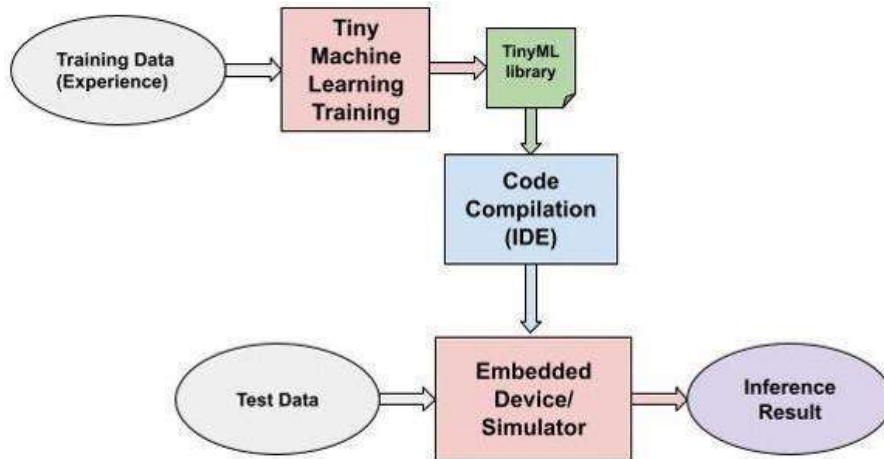


Fig. 4. The Edge AI tool stack

3.3 Machine Learning Process

To assist in the process of training each of the data samples had an associated label. The raw data was first undertaken through a preprocessing stage in which it was cleaned and formatted to ensure that the used set is representative. Preprocessing was followed by the extraction of features from the dataset. Each extracted feature was associated with a label then saved in separate files. Backpropagation was then used to train the model using different features and the associated labels, with this technique several iterations were applied to update model parameters to increase the chances of predicting a label until an acceptable performance was reached before deployment to a device. Fig. 5 shows the machine learning process.

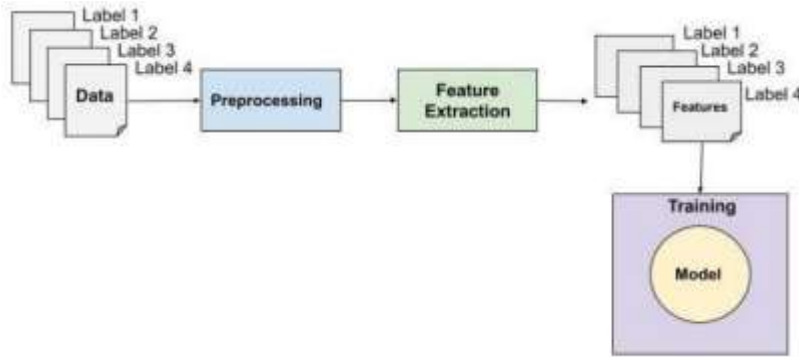


Fig. 5. Machine Learning Process

3.4 Synthetic Data Generation

To increase data sets in cases where there are low datasets, synthetic data was generated. The steps for synthetic data generation are presented in Fig 6. It starts with a sample raw data that is uploaded into the synthetic data platform in a CSV format and submission acknowledgment given to the user. This is followed by the provisioning step in which free computing resources are allocated, thereafter the encoding process in which data is transformed is done. After encoding, a deep neural network model is trained and on completion used to randomly draw a synthetic dataset. After generation, the accuracy and privacy of the data are analyzed and a quality assurance report is generated.

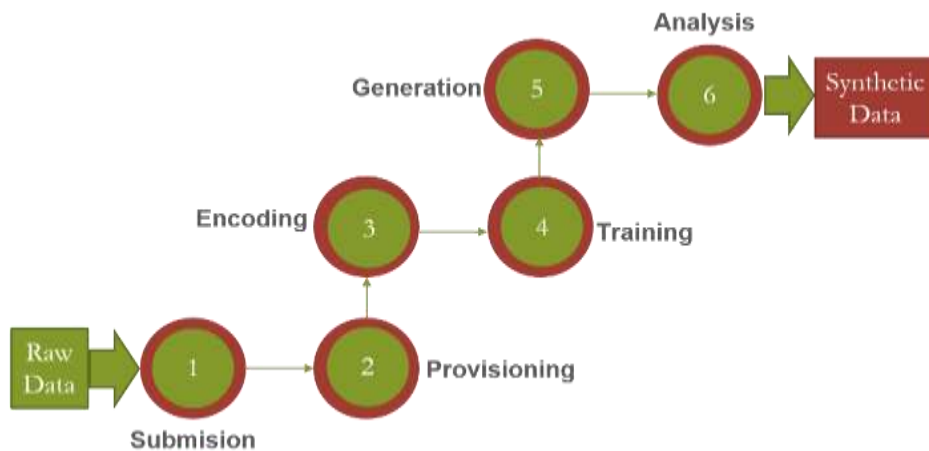


Fig. 6. Synthetic Data Generation Steps

3.5 Anomaly detection

For anomaly detection, we used a K-means anomaly detection learning block. With this, the selected features are taken and normalized with a standard scaler. A K-means clustering is then run over this feature space with the number of clusters provided and for all clusters found the center of the cluster is determined, and the radius. The classification data is again normalized, the closest cluster to the incoming data checked with the anomaly score being the distance to the closest cluster.

3.6 Use of Dynamic Inference

During experimentation, dynamic inference [20] was applied. This enabled the NN to adapt their parameters or structures to the given inputs, unlike with static models which parameters and computational graph are fixed at the inference stage. This leads to computational efficiency, adaptiveness, and accuracy. In our study, we used a sample-wise adaptive model that processes each sample by first checking for anomalies then after the NN classification. Fig. 7 shows the dynamic inference process.

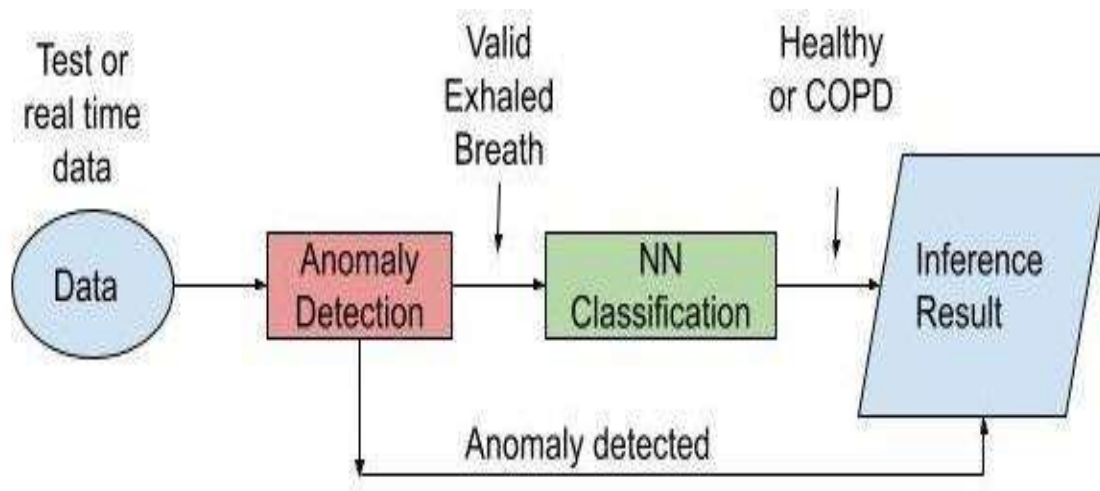


Fig. 7: Dynamic Inference process

3.7 System Design Method

The Agile software development method was selected to the development of the embedded system given its evolutionary and iterative nature. This allowed the improvement of the system and the edge AI model with each iteration. With this technique it was easy to decide on areas of improvement with each iteration. This made the development process rational and quick allowing the addressing of most pressing issues immediately. Figure 8 shows the steps involved in the agile development method

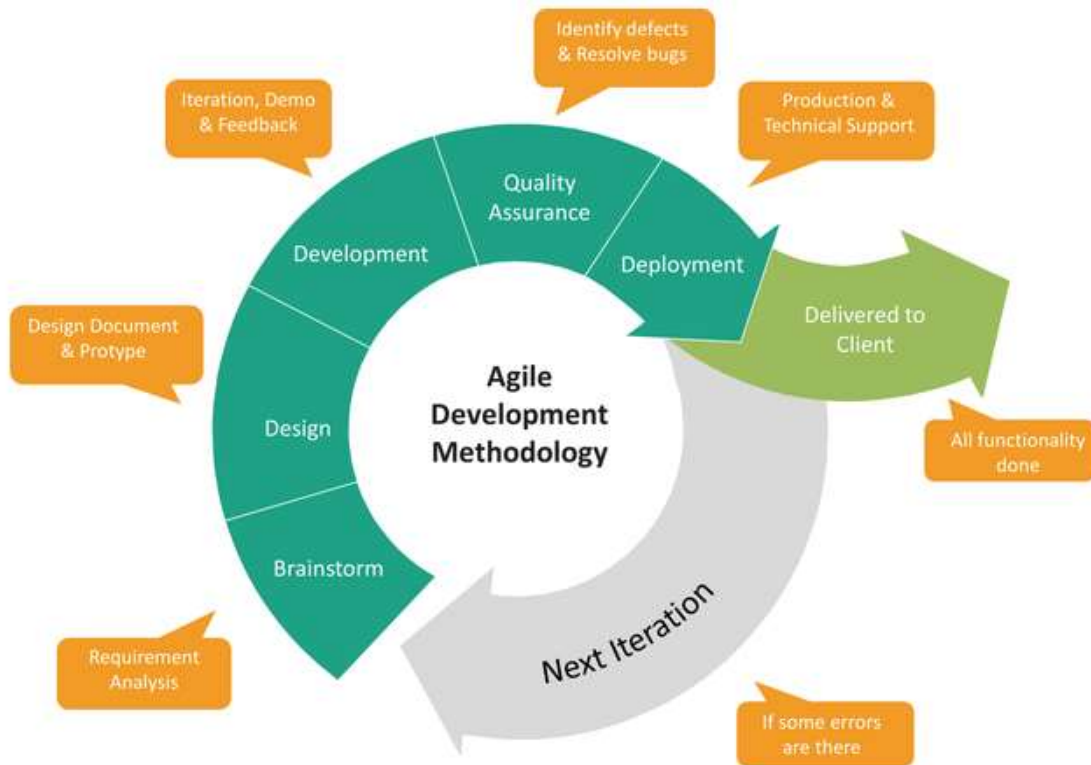


Fig 8: Agile software development steps

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

In this chapter, the material and methods used in the study are given. The first section gives the high-level system architecture. This is followed by a detailed embedded System-level design showing the system block diagram, Original Equipment Manufacturer (OEM) components, system Program Description Language (PDLs), use case diagrams, and system flow charts. Lastly, the modelling and layout of the embedded system are given.

4.1 System Architecture

4.1.1 High-level System Architecture

Fig. 9 presents the high-level operational context of the solution. A given person breathes into a tube that canalizes the exhaled air towards VOC sensors. The VOC sensors then translate the detected breath sample into a digital format for further processing and analysis. An edge AI model is then used to first check for any anomalies in the collected sample and if there are no anomalies inference is done to determine the respiratory health status of the person. A status notification is then being given via an LCD screen and LEDs with a green LED is lit when healthy and a red LED lit if infections are detected. Additionally, a provision is included for sending feedback to a user's mobile phone via Bluetooth technology. A heating pad is used to maintain the humidity within the sensing unit with a humidity sensor being used to monitor the humidity levels from time to time.

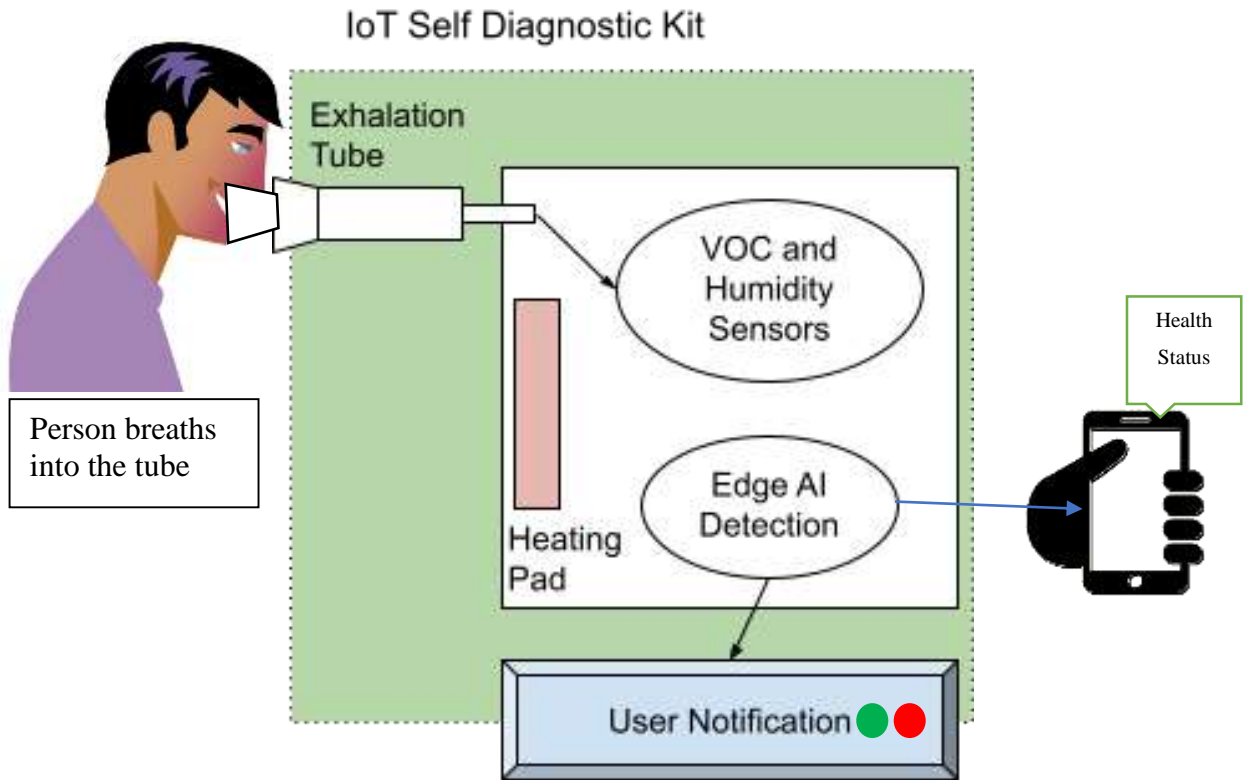


Fig. 9: High-level System Architecture

4.1.2 IoT System Architecture

The three-layer IoT Architecture [40] was selected for the solution. Fig. 10 shows the different layers of the system architecture and their functionalities

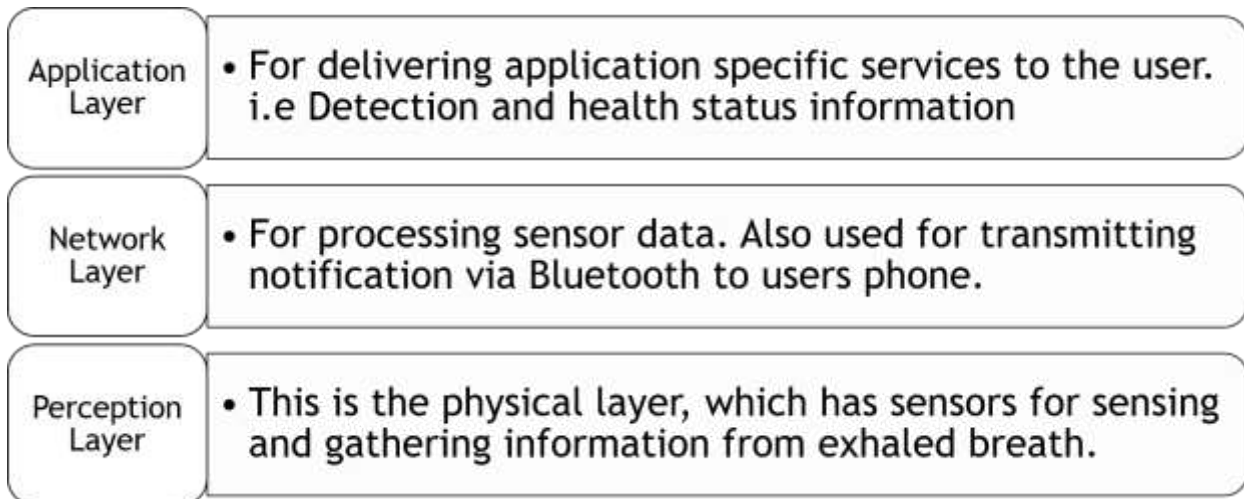


Fig. 10: IoT System Architecture

4.2 Embedded System-Level Design

4.2.1 System Block Diagram

As shown in Fig. 11, our embedded system is made up of 4 commercial gas sensors that can detect volatile organic compounds in exhaled breath (MQ-3 Alcohol Sensor, MQ-138 Formaldehyde Sensor, MQ-137 Ammonia Sensor, and MQ-135 Air Quality Sensor), an ARM Cortex 4 MCU for executing the inference locally, LEDs and an LCD screen to inform the user about the inference results in a user-friendly way. Furthermore, the system is equipped with a short-range wireless personal area network technology such as Bluetooth Low Energy (BLE) to potentially link the embedded device with the user's mobile phone.

Even though the proposed system may be powered by mains, our design relies mainly on a battery power supply to facilitate its portability and use anywhere, anytime.

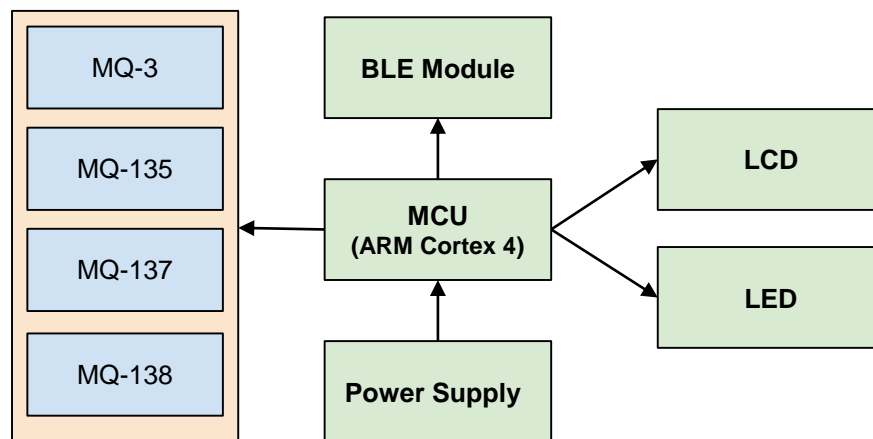


Fig. 11. System Block Diagram

4.2.2 Hardware Components

The main hardware components for the prototype include the following;

- Arduino Nano BLE sense
- MQ-135 Sensor
- MQ-137 Sensor
- MQ-3 Sensor
- MQ-138 Sensor
- LCD
- RGB LED

A. Arduino Nano BLE sense

The Arduino Nano 33 BLE sense is built upon the nRF52840 microcontroller. It runs on ARM Mbed OS making it appropriate for solutions that involve embedded machine learning. In addition, it has an integrated Bluetooth low energy module and a variety of sensors including temperature and humidity sensors [41]. Fig. 12 shows the pin layout for the board.

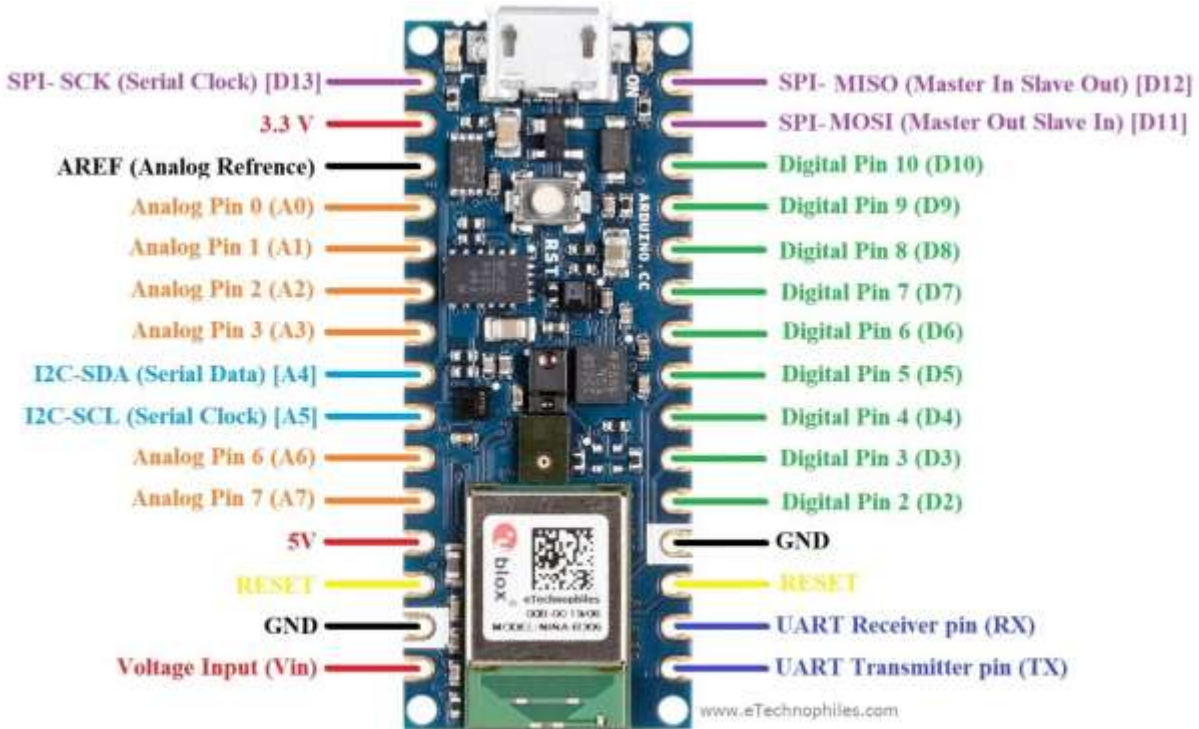


Fig. 12: Arduino Nano BLE Sense pin layout

B. MQ-3 Gas Sensor

The MQ-3 module [42] can detect Benzine, Alcohol, Hexane, CH₄, Carbon monoxide, and liquid petroleum gas. SnO₂ is the sensitive material in an MQ-3 gas sensor, whose conductivity reduces based on the cleanliness of the air. When the sensor detects a targeted alcohol gas, the sensor's conductivity rises as the concentration of the gas increases. There is the resistance across an A and B inside the sensor which varies on detection of a targeted gas. The targeted alcohol concentration is measured by measuring this resistance. The concentration of alcohol is inversely proportional to the resistance. The sensor and load resistor form a voltage divider, and the lower the sensor resistance, the higher the voltage reading will be. Fig. 13 below shows the sensor module pinout

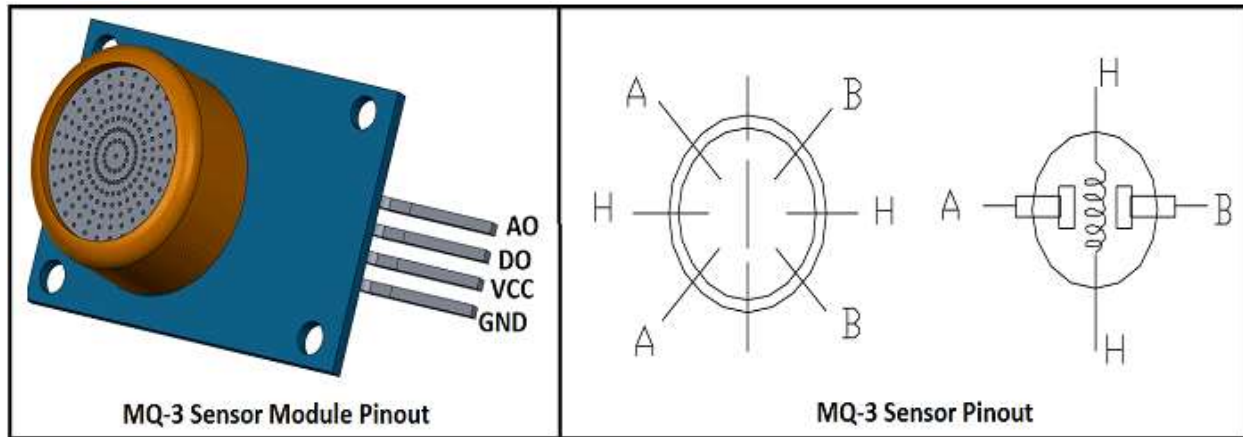


Fig. 13: MQ-3 Module Pinout

C. MQ-135 Gas sensor

The MQ-135 Gas sensors [43] are used to measure air quality. They are suitable for measuring or detecting Alcohol, NH₃, Smoke, NO_x, CO₂, Benzene, The MQ-135 sensor module has a digital Pin making it possible to operate without a microcontroller especially when trying to detect a particular gas. However, if there is a need to measure the concentration of a gas in PPM the analog pin is used. MQ-135 gas sensor applies SnO₂ which has higher resistance in the clear air as a gas-sensing material. The resistance of the gas sensor increases with an increase in polluting gases.

Fig. 14 shows the pin layout for the MQ-135 sensor

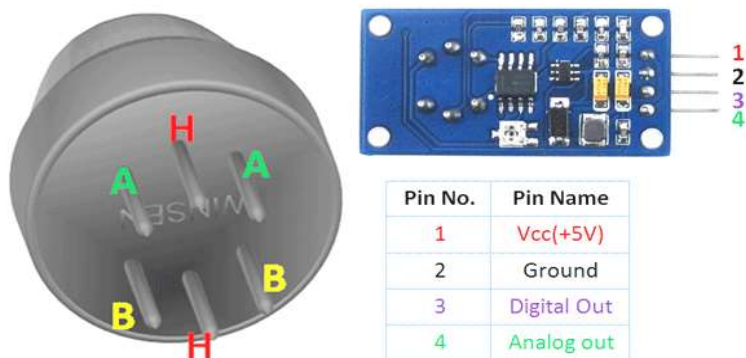


Fig. 14: MQ-135 Sensor pin layout

D. MQ-137 Gas Sensor

The MQ-137 Gas sensor [43] can measure or detect the concentration of Carbon Mono-oxide (CO) and Ammonia (NH₃). The MQ-137 sensor module has a digital Pin like the MQ-135 sensor making it possible to operate without a microcontroller especially when trying to detect a particular gas. However, if there is a need to measure the concentration of a gas in PPM the analog pin is used. The resistance of the gas sensor increases with an increase in polluting gases. Figure X shows the pin layout for the MQ-135 sensor. Fig. 15 shows the sensor pin layout.

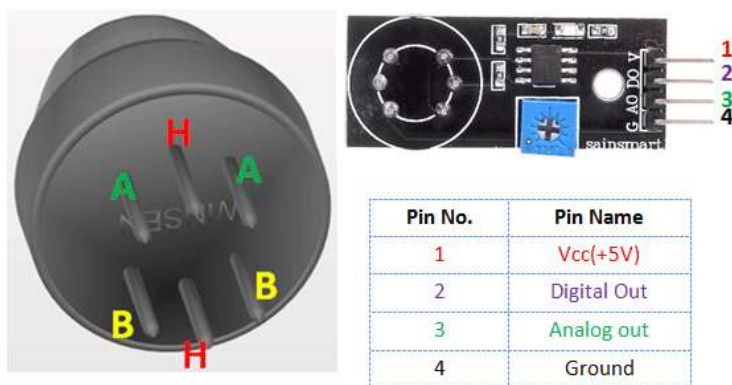


Fig. 15: MQ-137 Sensor pin layout

E. MQ-138 Gas Sensor

The MQ-138 [44] can detect the levels of VOC in the air and air quality. SnO₂ is used a sensitive material of the gas sensor given the ability to change conductivity based on air quality. the sensor's conductivity increases with the gas concentration rising when VOCs are detected in a gas. The change of conductivity can be converted to correspond to the signal output of the gas concertation by using a simple circuit. The MQ138 gas sensor has a high sensitivity to acetone, toluene, alcohol, methanol, with the capability of monitoring hydrogen and other organic vapour. Fig. 16 show a representation of MQ135 gas sensor



Fig. 16: MQ-138 Sensor

F. LCD Module

An LCD screen is a module for electronic displays. Liquid crystal is used to produce visible images. The LCD comes in various shapes and sizes for example 16×2 LCD can display 2 lines with a maximum of 16 characters per line. In this LCD each character is displayed in a 5×7-pixel matrix [45]. Fig. 17 shows a 16x2 LCD pinout.

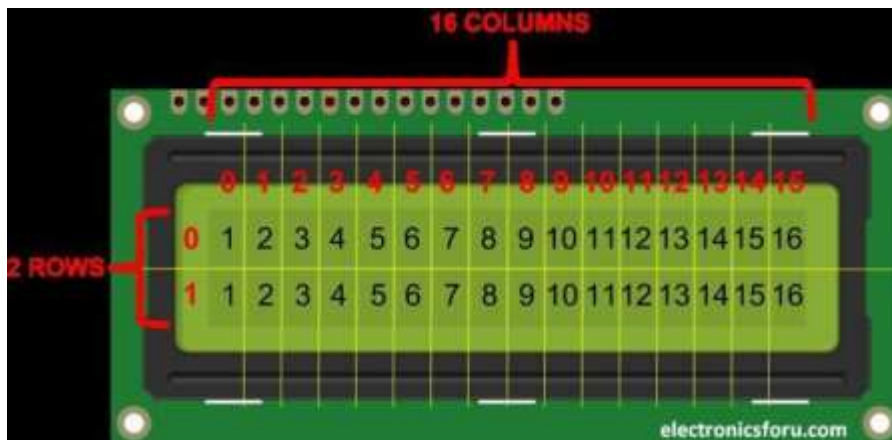


Fig. 17: 16X2 LCD pinout diagram

4.2.3 System PDL

Program description language (PDL) is free format English like text and was used to describe the flow of control and data in the system. The program is initiated when device is turned ON with the initialization of variables and a display of welcome messages. A button rise ISR triggers the collection of breath samples while a button fall ISR triggers the anomaly detection and classification of the samples followed by a use notification on health status. A collection of keywords is used when writing the PDL to describe the operation of a program in a logical and stepwise manner as outlined below.

```
BEGIN  
  Initialize Variables  
DOFOREVER  
  CALL WELCOME  
  Register ButtonRise_ISR  
  Display Instruction  
ENDDO  
END
```

```
BEGIN/WELCOME  
  Display Device Brand  
  Display Welcome  
  Wait 5 Seconds  
  Clear LCD  
END/WELCOME
```

```
BEGIN/BUTTONRISE_ISR  
  DO WHILE button is pressed  
    Display Instruction  
    Display 10 to 1 countdown  
    Sense breath VOC  
    Display Sampling Notification  
  ENDDO  
  Register ButtonFall_ISR  
END/BUTTON_ISR
```

```
BEGIN/BUTTONFALL_ISR  
  Check sample for Anomaly  
  IF anomaly detected THEN  
    Display Anomaly Notification  
    Light Blue LED  
  ELSE  
    Run Classifier  
    Display health status  
    Send health status notification  
    IF person Healthy THEN  
      Light Green LED  
    ELSE  
      Light RED LED  
    ENDIF  
  ENDIF  
END/BUTTONFALL_ISR
```

4.2.4 System Flow Charts

The flow charts for various sub-functions of the system are shown in Fig. 18. There is a main function with three sub routines; welcome, button rise ISR and button fall ISR. The main function initializes variables and then calls the welcome function that displays the welcome messages. Thereafter, the main function listens for button interrupts. The button rise ISR triggers the sampling process for exhaled breath while the button fall triggers the prediction model for the system.

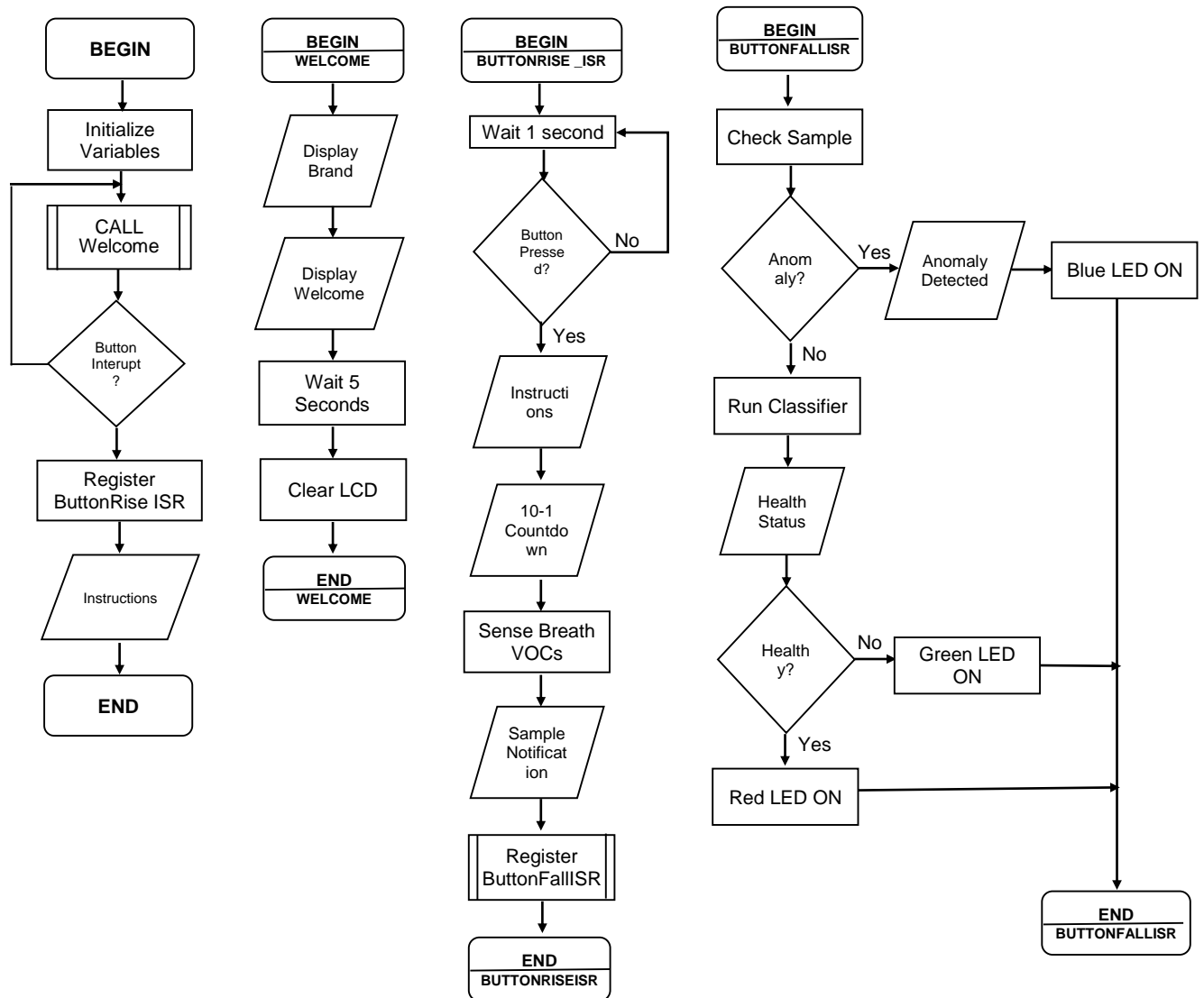


Figure 18: System flow charts

4.2.6 Use Case Diagram

Fig. 19 shows the use case diagram for the system. The use case diagram presents a summary of the system's users (often referred to as actors) details and their anticipated interactions with the system. The main identified actors were the user, the system administrator and the embedded system.

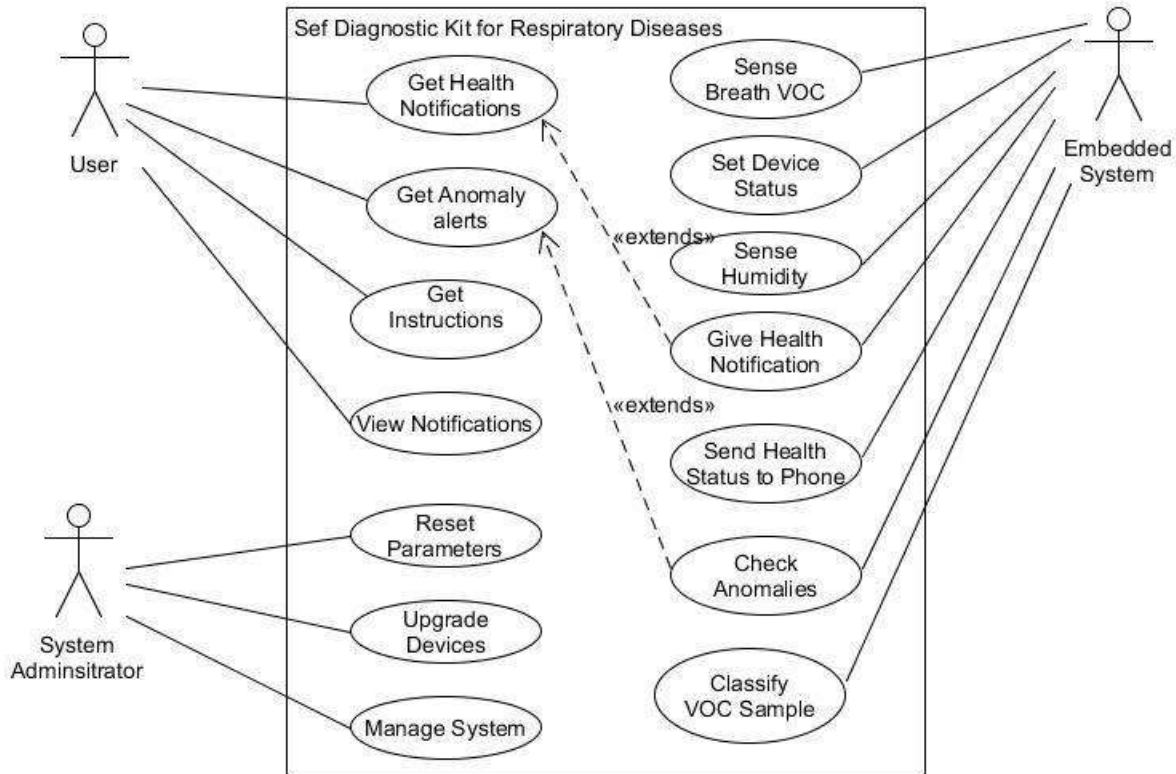


Fig. 19: Use case diagram

4.3 Edge AI model training architecture

The architecture for training our edge AI model is presented in Fig. 20. Raw data is taken into the pipeline and slashed into smaller windows, signal processing blocks are then used to for feature extraction, and then a learning block is used for the classification of the new data. For some inputs the same values are always returned by the Signal processing blocks and are used to make processing of raw data easier. The learning blocks learn from past experiences. The learning block has two parallel training processes, one for detecting abdominal observations and the second one for classifying observations as healthy against COPD or not.

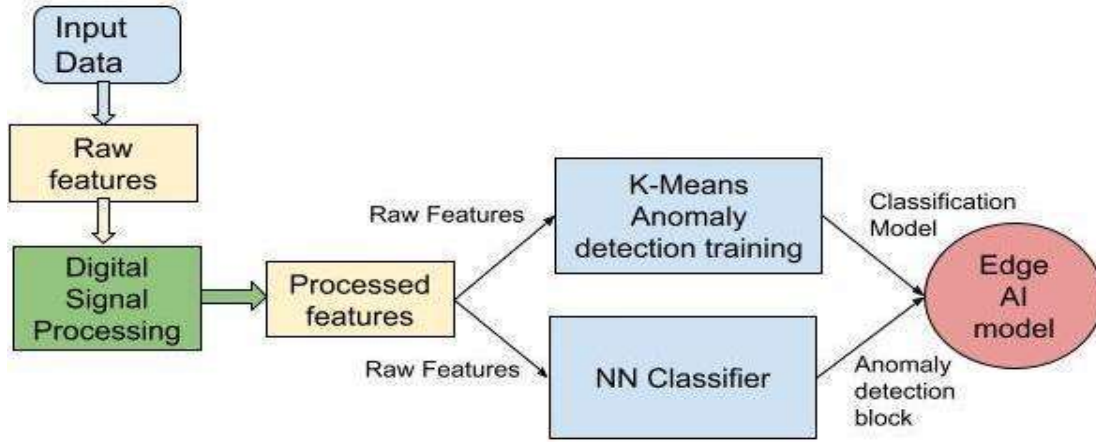


Fig. 20. Architecture for Training our COPD Edge AI Model

4.4 System Simulation

The hardware configurations were done on STM32Cube IDE. Figures 21 and 22 show the clock configuration and pinout for the selected STM32 board respectively

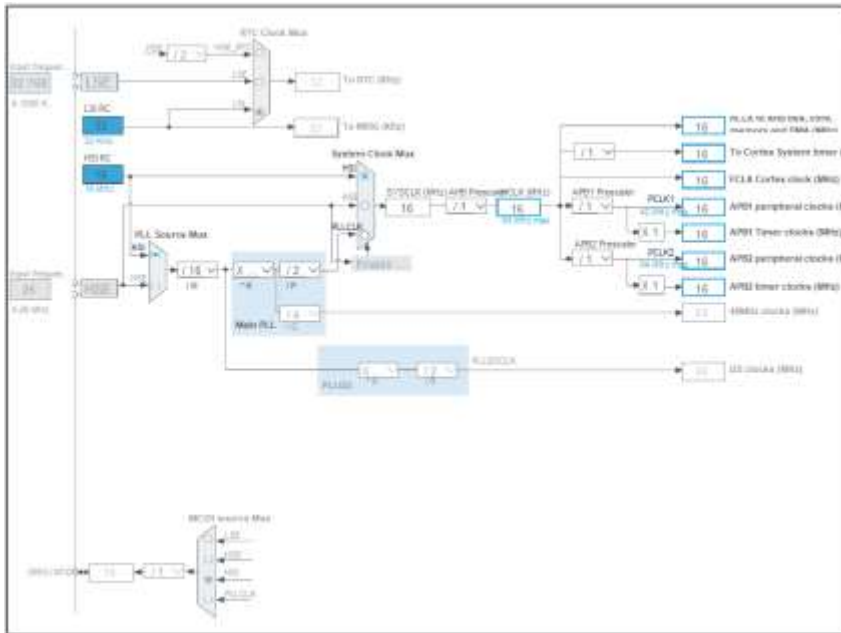


Fig. 21: STM32F401CEUx Clock Configuration

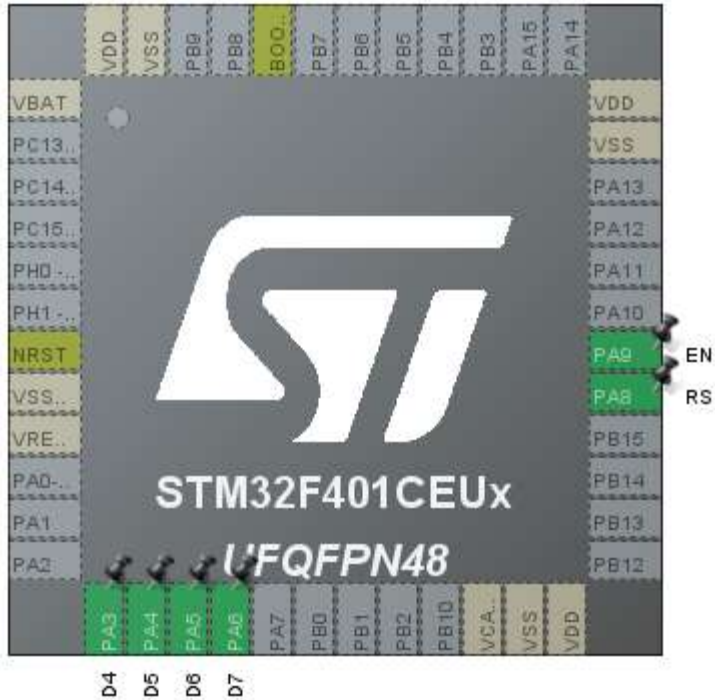


Fig. 22: STM32F401CEUx Board pinout

Fig. 23 shows our simulation model on the Proteus design suite which enables the rapid design testing and layout of printed circuit boards based on a combination of ease of use and a powerful embedded-compliant feature set. Two options of displaying the results on LCD and virtual terminal were used for the simulation. Due to the unavailability of the required sensors on the Proteus design suit. Data was given through an input file on an SD card to represent data collection from sensors.

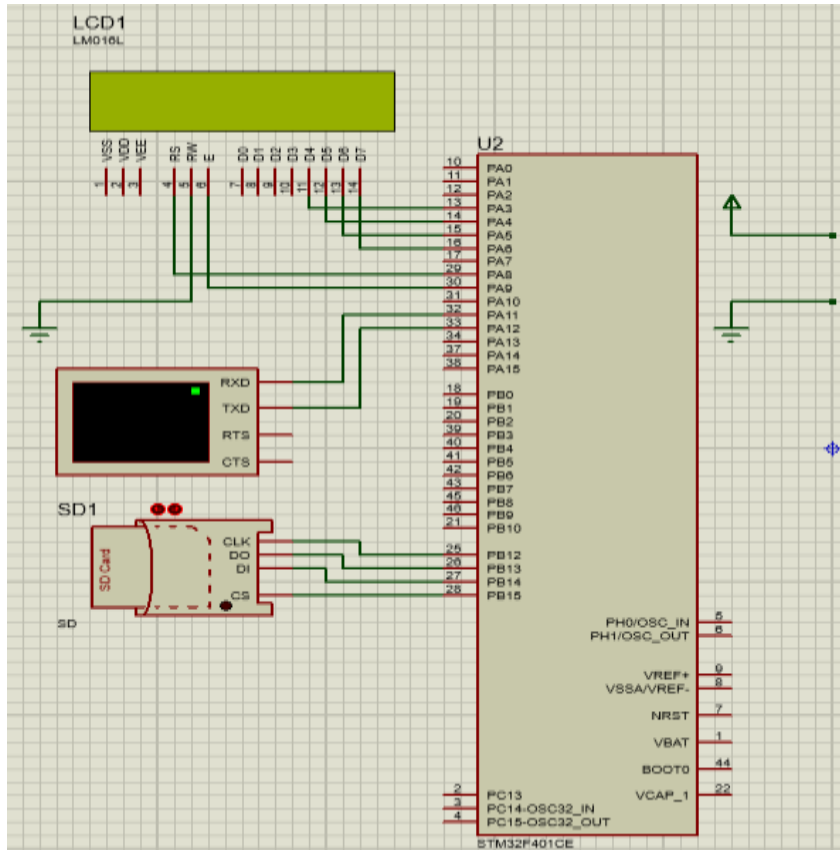


Fig. 23. Proteus Simulation Model

4.4 Embedded Device Set-Up

For the purposes of prototyping the MQ135 was used. The hardware components were connected as follows:

A. MQ135 Sensor

This sensor has 4 pins that were connected as follows:

- 5V to the Arduino 5V pin, it supplies power to the module.
- GND connected to the GND pin on the Arduino, this is the ground pin.
- DOUT was not used.
- AOUT connected to A2 pin of the Arduino, this is the Analog output connected.

B. LED

Connected the longer pin of RGB led is the cathode was connected to the GND of Arduino Nano 33 BLE sense with the remaining three pins of the RGB led being connected to pins 13, 12, 11 of Arduino through a 220 ohm resistors. The resistors were used for preventing the excess amount of current from flowing through the RGB led.

C. LCD

The LCDs have a parallel interface; this means that the number of interface pins have to be manipulated by the microcontroller at once to control the display. The following pin make up the interface:

- RS: A register selects pin that controls wherein the LCD's memory
- R/W: A Read/Write pin that selects writing mode or reading mode
- Enable: An Enable pin for enabling writing to the registers
- Data Pin (D0 -D7): The state of the pins are either high or low and are the bits that are written or read from the registers
- Others include power supply pins (+5V and Gnd) used to power the LCD, a display contrast pin (Vo) that is used are used to control the display contrast, and LED Backlight pins.

When wiring the LCD screen to the Arduino board, the pins were connected as follows:

- RS pin to digital pin 8 of the Arduino Nano 33 BLE sense
- The LCD Enable pin to digital pin 7 pin 8 of the Arduino Nano 33 BLE sense
- D4 pin to digital pin 6 pin 8 of the Arduino Nano 33 BLE sense
- D5 pin to digital pin 5 pin 8 of the Arduino Nano 33 BLE sense
- D6 pin digital pin 4 pin 8 of the Arduino Nano 33 BLE sense
- D7 pin digital pin 3 pin 8 of the Arduino Nano 33 BLE sense
- R/W pin GND pin 8 of the Arduino Nano 33 BLE sense
- VSS pin GND pin 8 of Arduino Nano 33 BLE sense
- VCC pin 5V pin 8 of the Arduino Nano 33 BLE sense
- LED+ to 5V of the Arduino Nano 33 BLE sense through a 220-ohm resistor
- LCD LED- to GND Additionally, wired a 10k pot to +5V and GND, with it's wiper (output) to LCD screens VO pin (pin3).

Fig 24 shows the embedded system image showing the different components

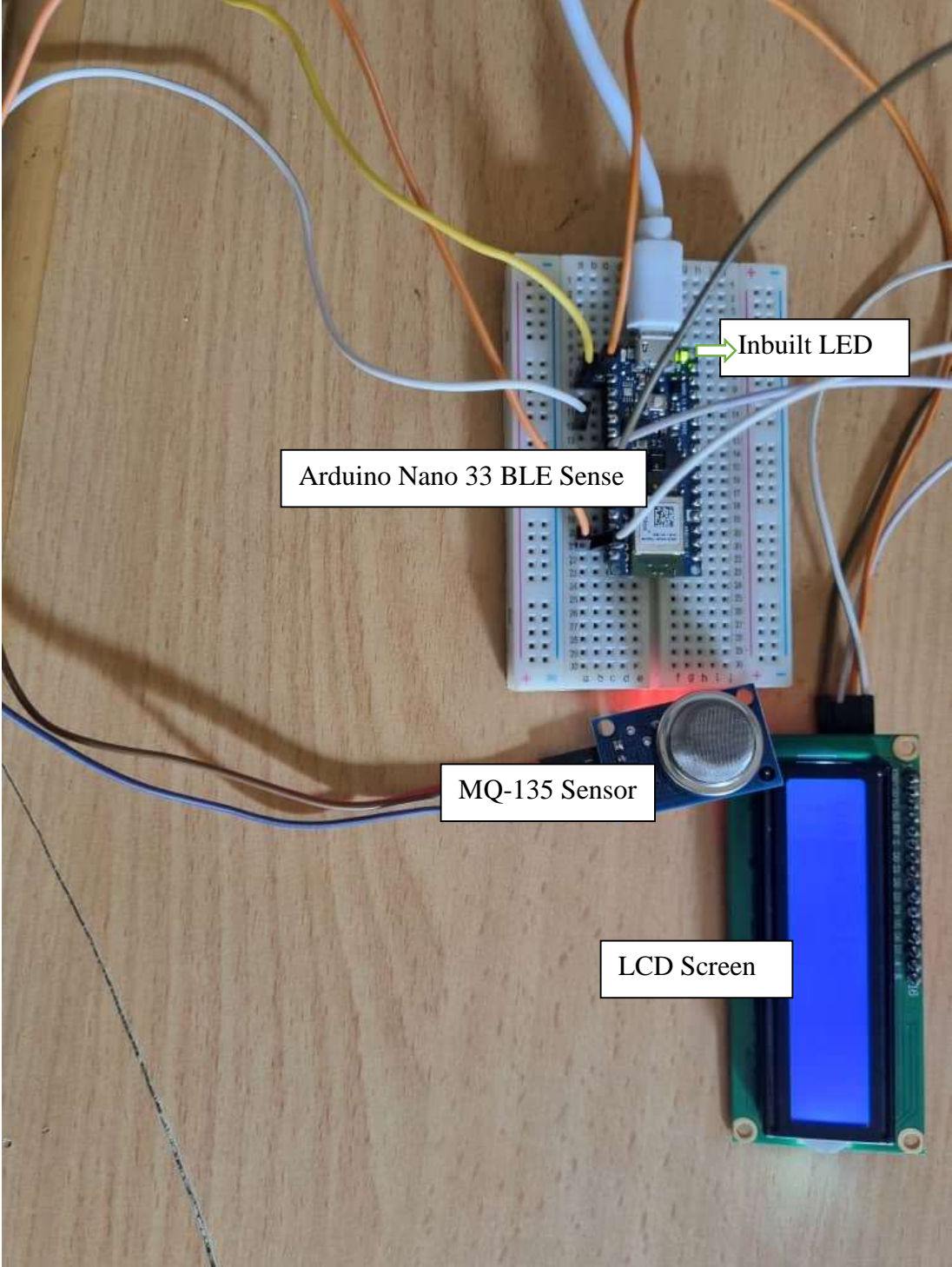


Figure 24: Embedded system prototype

CHAPTER 6

SYSTEM RESULTS AND ANALYSIS

The evaluation of our rapid prototyping tool stack has experimented in the use case of predicting COPD disease from exhaled breaths. This section first presents an evaluation using open data sets in which it introduces the input dataset, then presents our edge AI Tiny Model training process before describing the inference simulation results on a virtual and real embedded board. The second section presents experimentation of the use of synthetic data in cases of low datasets for the prediction of COPD.

5.1 Edge AI model

5.1.1 Open datasets

The open dataset of COPD data used in our experiment has been found in, [8]. The dataset was collected using a device with an array of 8 gas sensors (SP-3 VOC sensor, MQ-3 Alcohol Sensor, TGS 822 Organic Solvent Vapor Sensor, MQ-138 Formaldehyde Sensor, MQ-137 Ammonia Sensor, TGS813 Combustible Gases Sensor (Methane, Propane, Iso-butane), TGS800 Carbon Monoxide Sensor, and MQ-135 Air Quality Sensor) that captured the respective gas measurements in exhaled breath. Figure 5 gives a sample of the collected data. The data collection campaign involved 10 healthy people, 20 people suffering from COPD, and 10 samples from the air. Two breath samples were taken from each participant making a total of 60. Each observation had 500 samples at a sampling rate of 2ms with 16 observations per person. Figures 25 and table 2 give a representation of the sensing device and sample collected respectively.

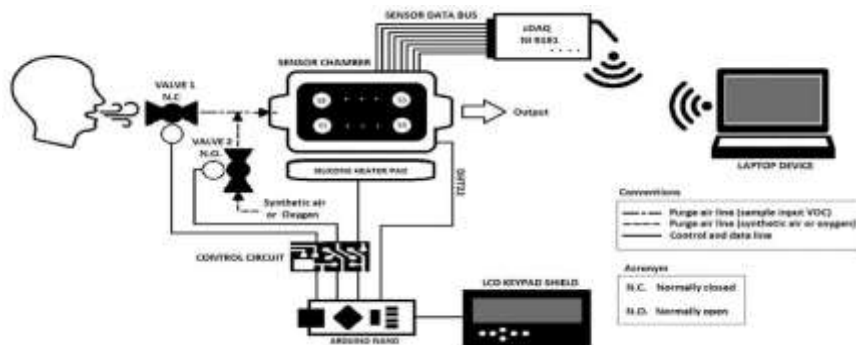


Fig. 25. Data collection device setup. The person breathes into the sensing unit. Data is collected by the sensors and sent wirelessly to a computer.

Table 2: Sample collected data. Data from all the 8 sensors recorded on a CSV format at a sample rate of 2ms

SP3	MQ3	TGS822	MQ138	MQ137	TGS813	TGS800	MG135
0.224	0.259	0.354	0.516	0.559	0.058	0.14	0.277
0.224	0.259	0.354	0.516	0.559	0.058	0.14	0.277
0.224	0.259	0.354	0.516	0.559	0.058	0.14	0.277
0.224	0.259	0.354	0.516	0.559	0.058	0.139	0.277

5.1.2 Training Steps

The first step is to send data to the Edge impulse service. Data was uploaded using a preformatted JSON format according to their specifications. Two sets of data were uploaded, a dataset of samples taken from healthy persons labeled as healthy and a dataset of samples from people who were diagnosed to be suffering from COPD labeled as COPD. Fig.26 shows the raw data plot for COPD.

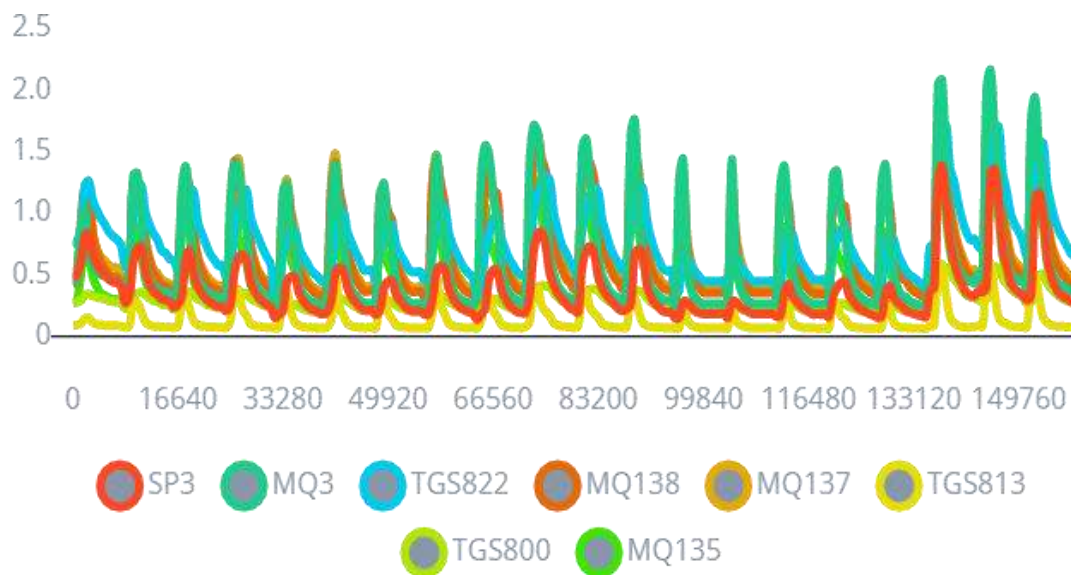


Fig. 26. Raw data - COPD

In a second step, features were extracted from the uploaded data before training. This is done to anticipate the fact that during the inference process, feature extraction on raw data by a deep learning model would require more processing power than available in embedded systems.

In the last stage, the observations are fed to 2 parallel TinyML training blocks; the first one being for detecting abnormal observations like air samples and the second being for inferencing the COPD disease.

5.1.3 Training parameters and output

As configuration, we set the model cluster count to 32 with the threshold value of 0.30. For the COPD inference, we used a Neural Network (NN) classifier composed of four layers with the input layer having 4,000 features, two dense layers of 20 and 10 neurons respectively, and an output layer of 2 features. The window size was set to 1000ms.

For both training runs, different numbers of training epochs were tested and obtained a good performance already at 50 epochs, with the learning rate set at 0.0005 and the minimum confidence rating at 0.60. A validation accuracy of 95.3% was achieved with a loss of 0.16%. Fig. 27 shows the confusion matrix for the validation set.



Fig. 27: Confusion matrix for the model on the validation dataset.

5.1.4 Inference on a virtual embedded board

The edge AI model developed in the previous section was first packaged as a TinyML library targeting CMSIS-PACK compliant STM32 boards. Using STM32Cube as Integrated Development Environment (IDE), the resulting library was integrated into our COPD disease prediction application as a library dependency before compiling and building the full application into an embedded executable to be deployed in our embedded simulation platform, which was proteus. For simulation, we used the STM32F401CE board.

In our embedded simulation environment, the same test data used during the training phase were read from a CSV file. The output inference results are written back to an output CSV file to be used to compare the accuracy of the results obtained during the training phase in the cloud. Fig. 28 shows the inference result of one test observation on an LCD in proteus respectively.

The comparison of inference simulation results and inference results obtained using the Edge Impulse cloud platform shows that the edge AI model achieves similar inference accuracy when running on embedded processors as well as on the cloud.

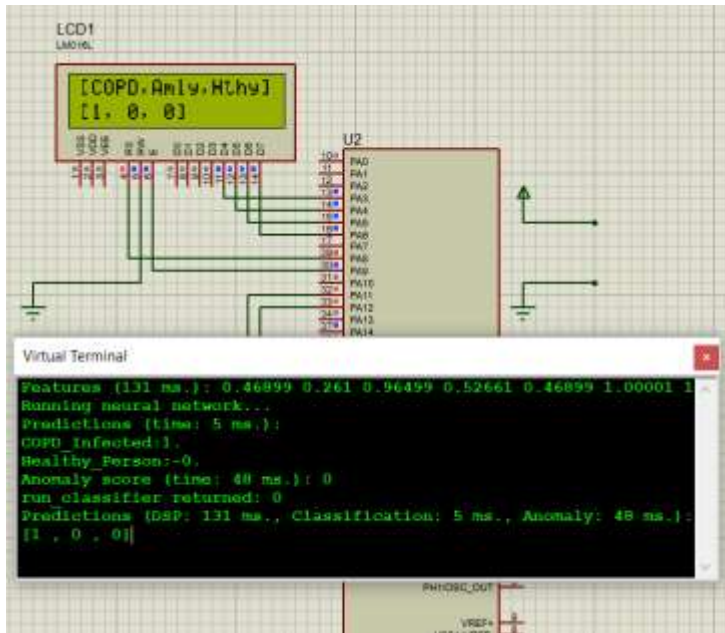


Fig. 28: COPD infected Sample Simulation Output on both the LCD and the serial monitor.

5.1.5 Inference on the real embedded board

The edge AI model developed in the previous section was first packaged as a TinyML library targeting Arduino boards. Using Arduino IDE as the Integrated Development Environment (IDE), the resulting library is integrated into our COPD disease prediction application as a library dependency before compiling and building the full application into an embedded executable to be deployed in our embedded prototype based on Arduino Nano 33 BLE Sense.

In our first inference experiment, the same test data used for inference in the cloud was applied for inference on the real board. Fig. 29 shows the inference result outputs from the embedded device.

```
anomaly score: -2.265
Edge Impulse standalone inferencing (Arduino)
run_classifier returned: 0
Predictions (DSP: 0 ms., Classification: 10 ms., Anomaly: 472 ms.):
[0.98047, 0.01953, -2.265]
  COPD_Infected: 0.98047
  Healthy_Person: 0.01953
  anomaly score: -2.265
Edge Impulse standalone inferencing (Arduino)
run_classifier returned: 0
Predictions (DSP: 0 ms., Classification: 10 ms., Anomaly: 472 ms.):
[0.98047, 0.01953, -2.265]
  COPD_Infected: 0.98047
  Healthy_Person: 0.01953
  anomaly score: -2.265
```

Fig. 28: Inference results on real board

This was followed by a collection of breath samples from a few volunteers who breathe into the prototype device and inference done to determine if they are healthy or not. Figure 29 shows a sample output from the embedded device. The sample result shows that the probability of the person being healthy was 84% with that of being infected at 16% hence the person is considered healthy and the LED lights green.



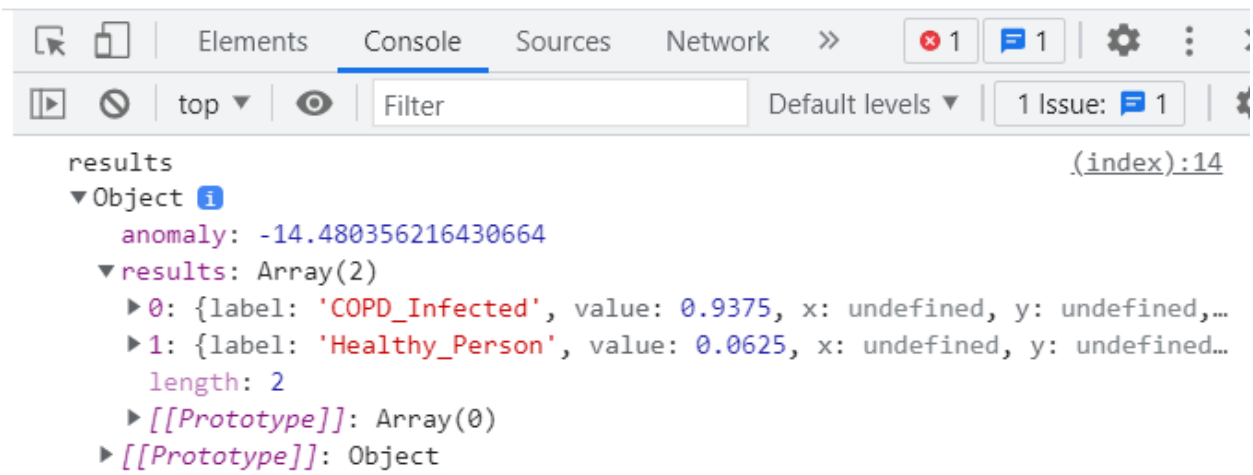
Fig 29. Inference output from embedded device

The comparison of inference simulation results and inference results obtained using the Edge Impulse cloud platform shows that the edge AI model achieves similar inference accuracy when running on embedded processors as well as on the cloud. Interestingly, our model successfully classified air samples as an anomaly. From the results, we conclude that our edge AI model is effective in predicting whether someone is infected by COPD or healthy from exhaled breath.

Anomaly detection helped in avoiding the risk of the edge AI model inference of an input that is not a valid exhaled breath. In our case, this enabled us to identify the air samples cluster that is different from the exhaled breath cluster with an accuracy of 100% on the available test dataset, even though further validation with other datasets will be needed.

5.1.6 Inference on local machines

The model was also deployed as a WebAssembly library. This packaged the learning blocks, signal processing blocks and configuration as a single package. This package was included in web pages or as part Node.js application in user devices. This allowed the running of the prediction model locally without the need for compilation. For running on webpages python 3 was required to be installed in the devices and needed code written as shown in the appendices. Figure 30 and 31 give the outputs from the browser and Node.js respectively.



```
results (index):14
▼ Object ⓘ
  anomaly: -14.480356216430664
  ▼ results: Array(2)
    ▶ 0: {label: 'COPD_Infected', value: 0.9375, x: undefined, y: undefined,...}
    ▶ 1: {label: 'Healthy_Person', value: 0.0625, x: undefined, y: undefined,...}
    length: 2
    ▶ [[Prototype]]: Array(0)
    ▶ [[Prototype]]: Object
```

Fig. 30: Browser prediction output


```
anomaly: 67.97413635253906,
results: [
  {
    label: 'COPD_Infected',
    value: 0.1015625,
    x: undefined,
    y: undefined,
    width: undefined,
    height: undefined
  },
  {
    label: 'Healthy_Person',
    value: 0.8984375,
    x: undefined,
    y: undefined,
    width: undefined,
    height: undefined
  }
]
C:\Users\Samson Ooko\Downloads\edge-ai-for-copd-(raw-data-model)-wasm-v19>
```

Fig. 31: Sample prediction output from Node.js

5.2 Experimentation on use of synthetic data

To prove the applicability of the use of synthetic data for cases of low data sets experimentation with synthetic data was done. This section presents the embedded machine learning process using the synthetic data for the prediction of COPD from exhaled breaths.

5.2.1 Input: Raw COPD Dataset

An open dataset for COPD [8] was used as the basis for the generation of synthetic data. This data set was collected by analyzing exhaled breath samples using an array of 8 gas sensors (SP-3, MQ-3, TGS-822, MQ-138, MQ-137, TGS-813, TGS-800, and MQ-135). Data was collected at the rate of 500 samples per second with 20 samples from healthy people and 40 samples from those infected with COPD. From every person, 8 samples were taken.

5.2.2 Synthetic Data Generation

Mostly AI [21], an online synthetic data generation tool was used to generate the synthetic exhaled breath data. Raw data was first uploaded to the platform in a CSV format. The number of processed subjects and the number of generated subjects were then set at 80,000. Table parameters were then set to include data from all 8 sensors and the number of training epochs set at a maximum of 100 with a batch size of 32 and a learning rate of 0.001. The generation process began following the synthetic data generation steps that included submission, provisioning, encoding, training,

generation, and analysis. The best model was reached at 74 epochs. The data was then synthesized and analyzed and a QA report was given.

5.2.3 Synthetic Data Quality Measurement

To ensure the quality of the generated synthetic data, an investigation was conducted to find out if the generated data was a statistical representation of the actual data and not an over-fitted copy of the real dataset. A holdout record was randomly selected from the actual dataset for reference because they have the same distribution but have not been seen during training. Even though the synthetic records should be as close as possible to the training data, their closeness to the holdout data should not be closer than expected since this would mean an information leak and not data pattern learning.

The similarity levels were quantified by first investigating the Identical Match Share (IMS) and ensuring that the matches of the synthetic data with the training data were not significantly different from the matches with the hold out data. Secondly, a check was done to ensure that occurrences in the actual data set were also present in the synthetic dataset by investigating the overall distribution of their distance to closest records (DCR). This was done by comparing the quantities of empirical distribution functions using statistical set tests. Lastly, the Nearest Neighbor Distance Ratio (NNDR) was used to normalize the distance to the closest record to the overall density within a data space region, by dividing it by the distance to the 2nd closest record. A check was done to ensure that the NNDRs for synthetic records are not systematically any closer than expected from holdout records. The similarity-based tests are presented in fig. 32.

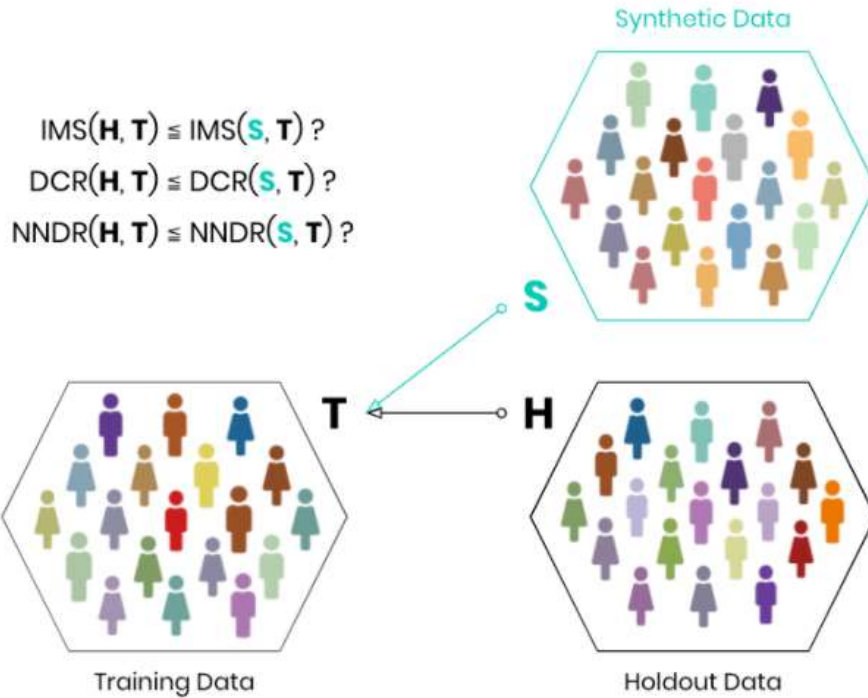


Fig. 32: similarity based and privacy tests [20]

5.2.4 Embedded ML model generation

The training process for the models followed the following steps. First, two different projects were created, one for the actual data model and the other for the synthetic data model. The data sets were first preformatted and a custom python script was used to create a JSON file for uploading to the edge impulse service. For each project, three sets of files were uploaded, one file with samples from healthy persons, one set with data from COPD-infected persons, and a test set that included 20% of raw data from each category. The holdout method as shown in fig. 33 was applied to randomly separate the uploaded 80% training data to 60% training set and 20% validation set.

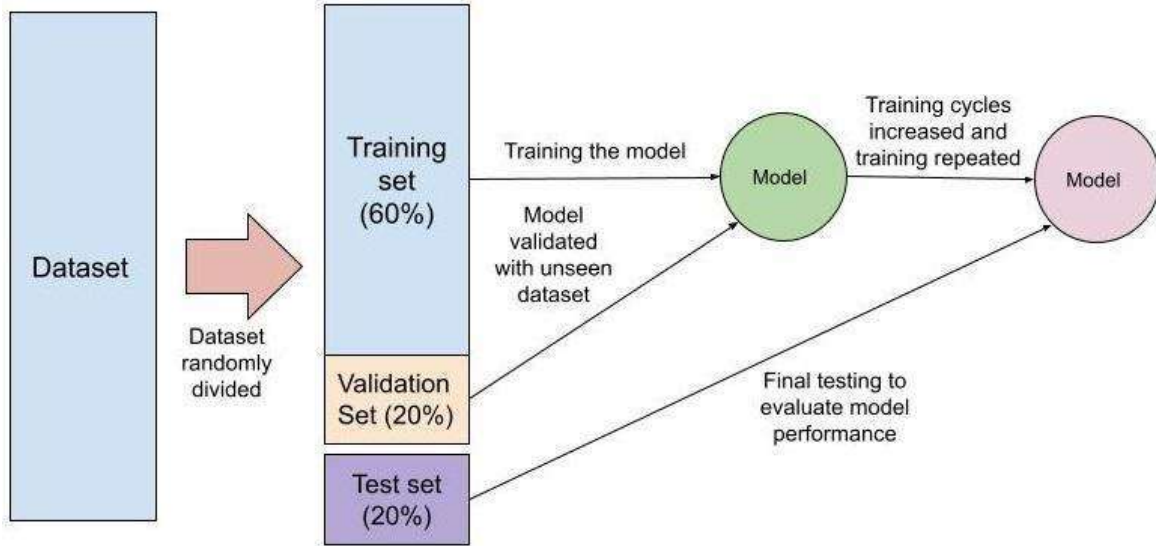


Fig. 33. Holdout Method. Data divided into 60% training set, 20% validation set, and 20% test set

The window size was set at 1000ms with a sampling rate of 2ms and a window size increase of 1000ms. A raw data processing block was then added with the input axis from all the 8 sensors selected. A Neural Network (NN) learning block and K-means anomaly detection learning block were added with raw data as input and 3 output features (Health person, COPD Infected, and anomaly). Fig. 34 shows a raw data sample plot.

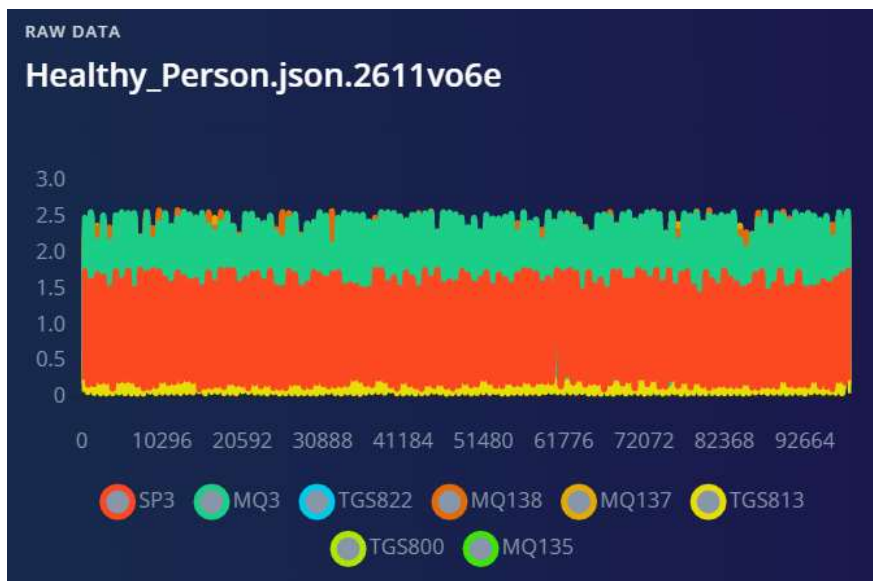


Fig. 34. Raw data sample

The features from the data were extracted before the model training this was done to reduce the amount of processing power that will be needed to generate the features in an embedded system. Fig. 35 shows a sample of the generated features.

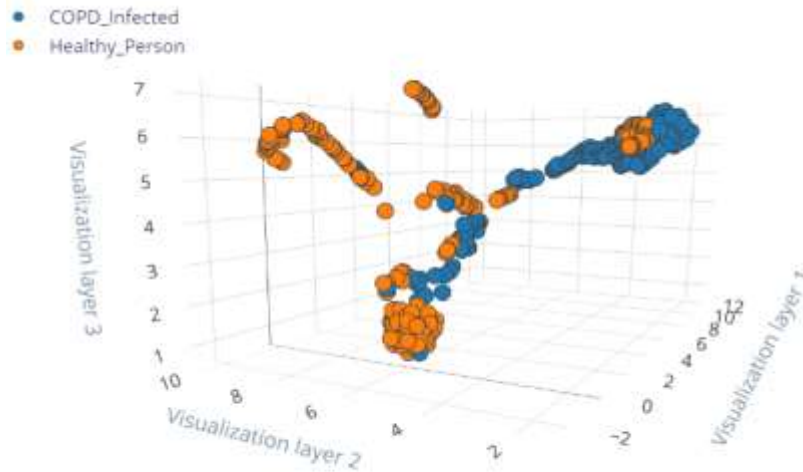


Fig. 35. Generated synthetic data features

The number of training epochs for both projects was set at 50 with a 0.0005 learning rate and 0.80 minimum confidence rating. The neural network architectures were each composed of four layers; an input layer with 4,000 features, 2 dense layers with 10 and 20 neurons respectively and an output layer with 2 features.

5.2.5 Synthetic data generation results

A validation accuracy of 94.2% was achieved with a loss of 0.11% for the model based on a synthetically enhanced data set. Fig. 36 shows the confusion matrix for the validation set



Fig. 36. Synthetic data model confusion matrix

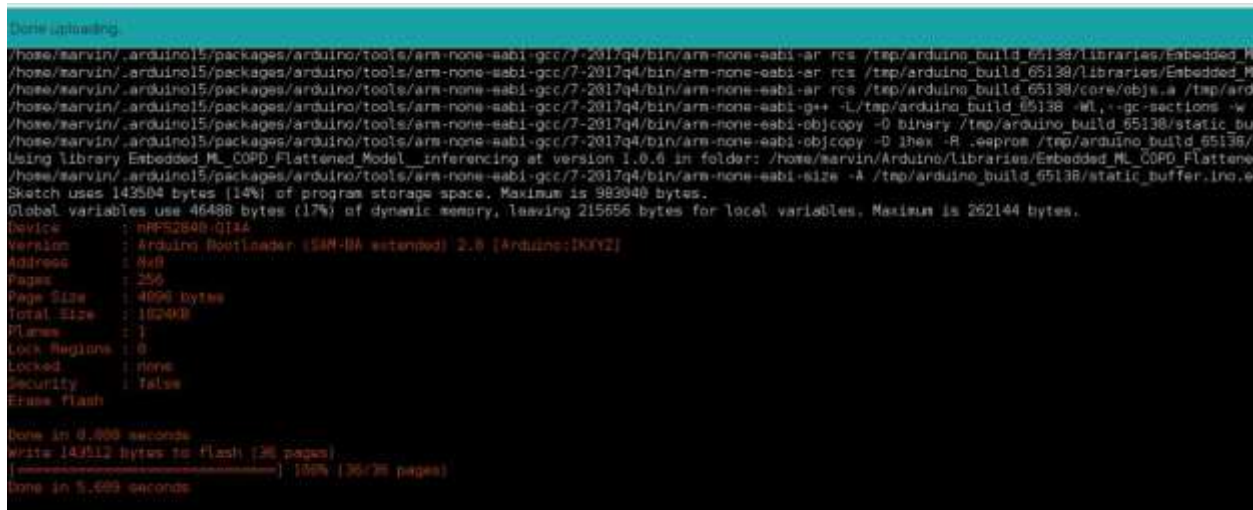
CHAPTER 6

DISCUSSIONS

After validating the effectiveness of our edge AI process in predicting COPD disease directly on end-user embedded devices, this section discusses different trade-offs between edge AI inference accuracy and embedded real-time resources. This trade-off analysis is a basis to understand the importance of precision edge AI development that consists of co-designing the embedded hardware and the TinyML model. This co-design will be deeply explored in our future research.

6.1 Real-time embedded processor specifications

According to edge Impulse estimation, the generated TinyML model performs the inference within 1ms and requires a memory of 1.5kB and 15.9kB respectively for peak RAM and ROM usage. On STM32cube compilation, it was observed that this tends to slightly increase with inference time up to 13ms, and memory of 18.7kB and 77.86 kB respectively for RAM and ROM. On the Arduino Nano 33 BLE sense board, the inferences time was 17ms with a dynamic memory usage of 46kB and program storage space of 143kB. However, the used resources on both the simulator and the embedded device are still less than 20% of the available resources in the embedded devices making the model appropriate for such devices. Fig. 37 shows the real-time embedded device resource usage.



```
Done uploading.
/home/marvin/.arduino15/packages/arduino/tools/arm-none-eabi-gcc/7-2017q4/bin/arm-none-eabi-ar rcs /tmp/arduino_build_65138/libraries/Embedded_M
/home/marvin/.arduino15/packages/arduino/tools/arm-none-eabi-gcc/7-2017q4/bin/arm-none-eabi-ar rcs /tmp/arduino_build_65138/libraries/Embedded_M
/home/marvin/.arduino15/packages/arduino/tools/arm-none-eabi-gcc/7-2017q4/bin/arm-none-eabi-ar rcs /tmp/arduino_build_65138/core/objs.a /tmp/ard
/home/marvin/.arduino15/packages/arduino/tools/arm-none-eabi-gcc/7-2017q4/bin/arm-none-eabi-g++ -L/tmp/arduino_build_65138 -Wl,--gc-sections -w
/home/marvin/.arduino15/packages/arduino/tools/arm-none-eabi-gcc/7-2017q4/bin/arm-none-eabi-objcopy -O binary /tmp/arduino_build_65138/static_bu
/home/marvin/.arduino15/packages/arduino/tools/arm-none-eabi-gcc/7-2017q4/bin/arm-none-eabi-objcopy -O ihex -R .eeprom /tmp/arduino_build_65138/
Using library Embedded_M_COPD_Flattened_Model_inferencing at version 1.0.6 in folder: /home/marvin/Arduino/libraries/Embedded_M_COPD_Flattene
/home/marvin/.arduino15/packages/arduino/tools/arm-none-eabi-gcc/7-2017q4/bin/arm-none-eabi-size -A /tmp/arduino_build_65138/static_buffer.ino.e
Sketch uses 143504 bytes (14%) of program storage space. Maximum is 983040 bytes.
Global variables use 46488 bytes (17%) of dynamic memory, leaving 215656 bytes for local variables. Maximum is 262144 bytes.
Device:
  Device:           : nRF52840-Q114
  Version:          : Arduino Bootloader (SAM-BA extended) 2.0 [Arduino:10X12]
  Address:          : 0x0
  Pages:            : 256
  Page Size:       : 4096 bytes
  Total Size:      : 1024KB
  Planes:           : 1
  Lock Regions:    : 0
  Locked:          : none
  Security:        : false
  Erase Flash:     :
Done in 8.808 seconds
write 143512 bytes to flash (36 pages)
[=====] 100% (36/36 pages)
done in 3.503 seconds
```

Fig. 37: Real Time embedded device resource usage

The projected on device performance for the edge AI model based on open dataset data and that based on synthetic data were the same. A variation was only witnessed depending on the preferred optimization for the NN classifier. For the quantized(int8) model a RAM usage of 5.4Kb, ROM usage of 92.9Kb, and a latency of 12ms are projected while the unoptimized (float32) model a RAM usage of 17.0Kb, ROM usage of 327.7Kb, and a latency of 54ms are projected. This shows that the optimized edge AI model will use less energy and therefore a longer device lifespan.

6.2 Impact of Preprocessing

When raw data was used without pre-processing and deep learning was used to learn features an accuracy of 96.9% was achieved with an estimated on device performance of 5.4Kb peak RAM, 94.9Kb ROM, and 12ms inference time. However, when the axis was flattened to a single value for each of the 500 samples per second accuracy of 95.3% is achieved with a better on device performance estimation of 1.5Kb RAM, 15.9Kb ROM, and inference time of 1ms. Fig. 38 shows the impact of feature reduction on accuracy and memory.

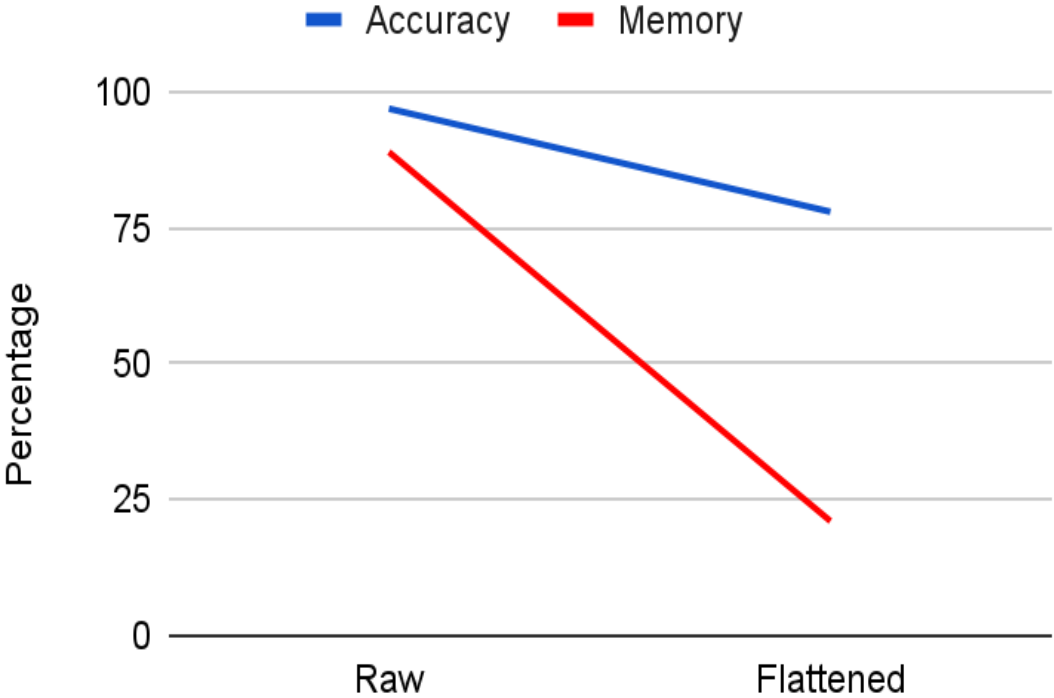


Fig. 38: Effect of feature reduction on accuracy and memory.

6.3 Importance of different sensors in inference accuracy

Another important question is to understand the importance of each of the 8 sensors on inference accuracy. Considering the worst case where only one sensor is used at run-time, the accuracy reduces to as low as 70% for a raw data-based model and 64.9% for a flattened based model. As the number of used sensors increases so does the inference accuracy at the cost of more RAM requirements. Fig. 39 shows the relative accuracy improvement for the raw-based model.

On device performance

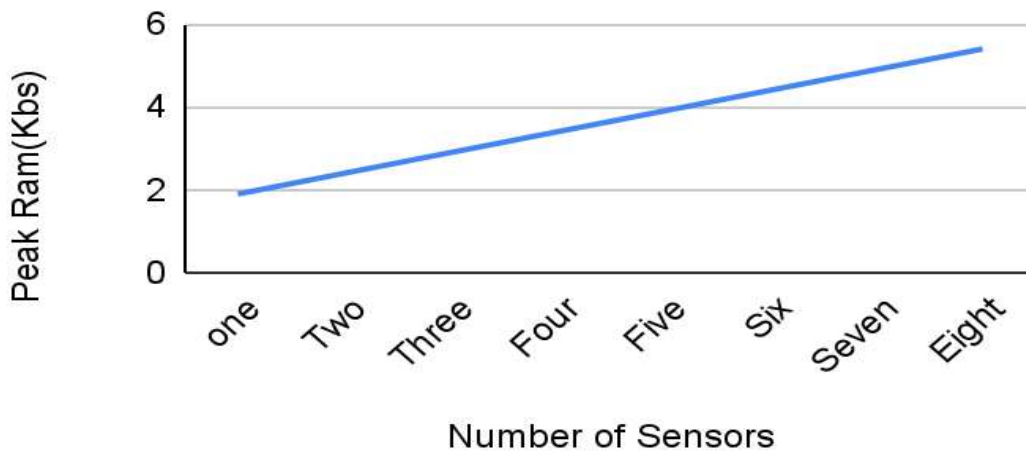


Fig. 39: On-device performance as the number of sensors is increased

Individually MQ sensors performed better than TGS and SP sensors on inference. Fig. 40 gives a comparison of individual inference performance by each sensor. Different combinations of 4 sensors were combined based on the accuracy achieved by individual sensors with a combination of MQ sensors giving the best result during inference. With this, we can conclude that if there were constraints one would confidently use an array of 4 MQ sensors and still achieve good inference results. Fig. 41 shows the inference performance of different sensor combination

Accuracy vs. Sensor

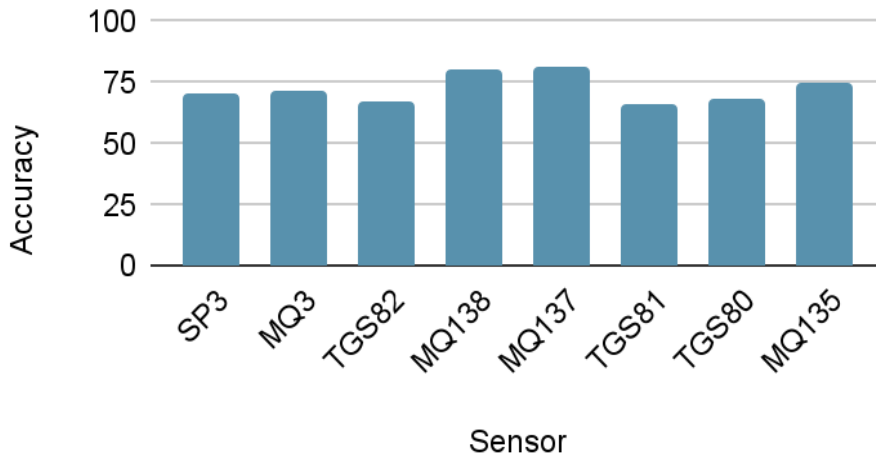


Fig. 40: Individual Sensor performance

Accuracy vs. Sensor Combinations

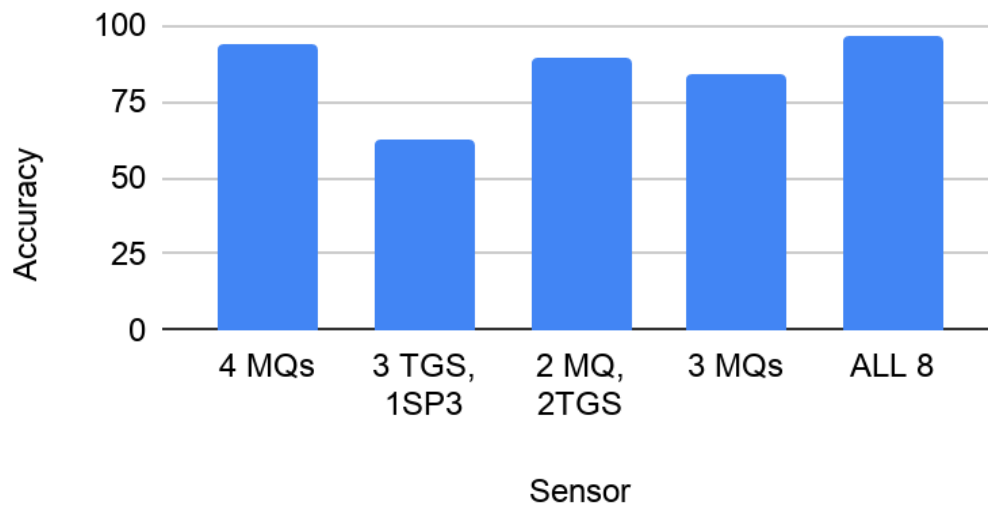


Fig. 41: Sensor combination performance

6.4 Applicability of Synthetic Data

The training accuracy of 96.9% was achieved with the model based on the open dataset with an accuracy of 94.2% being achieved on the model based on synthetic data. Figure 42 and 43 gives the confusion matrices for the two models. When test data was applied both models correctly predicted healthy and COPD-infected cases. The results prove that synthetic data can be relied upon for the training of models for the prediction of respiratory diseases.



Fig. 42: open dataset model confusion matrix



Fig. 43: Synthetic data model confusion matrix

6.5 Comparison to related works

A. Model Accuracy

The model performs better than that used to analyze the dataset in, [8]. In the previous study, the PCA method was used in a standalone computer to analyze the dataset with an accuracy of 92% being achieved. This is below 95.3% accuracy for our model.

B. Model Architecture and simulation

Our model is the first to be implemented at the Edge for the prediction of COPD and can be deployed in resource-constrained embedded devices. We are also the first to simulate a machine learning model from edge impulse on proteus.

C. Cost of the device

The device cost is approximated at USD 10,000 this is cheaper than the available commercial solutions which cost over USD 200. In addition, there is no need for transmission of data to the cloud which will automatically lead to savings on energy and data charges as shown in the table below. It is assumed that a person uses the device once a day.

	Active energy for data transmission via GSM	Costs
Our Edge based Solution	0	Purchase of device USD 100
A cloud based Solution	$2A \times 30 = 60$ A per month	USD 6 per month for connectivity Dr. consultation fees USD 10 per visit = 300 per month if visited daily Lab costs USD 10 per visit average = 300 per month Totals= 606

D. Portability for home use

Due to the fact that our device does not need any internet connectivity, it is portable for use at home irrespective of the availability of internet connection

6.6 Synthetic data generation results

From the holdout data, the Identical Match Share (IMS) was 1.9% with that of the synthetic data being 0.8%. This shows that the share of subjects within the synthetic data that matches an actual subject from the target data is not significantly bigger than the share that is to be expected when analyzing the target data itself.

From the holdout data, the 5th percentile of the Distance to Closest Record (DCR) was 0.6 with that for the synthetic data being equal to 0.8. This shows that the normalized distance for synthetic subjects to their closest actual subject within the target data is not significantly closer than the distance that is to be expected when analyzing the target data itself.

The Nearest Neighbor Distance Ratio (NNDR) for the 5th percentile of the holdout data was 0.6 and 0.7 for the synthetic data. This means the distance ratio between nearest and second-nearest records for synthetic subjects to their closest actual subject within the target data is not significantly closer than the ratio that is to be expected when analyzing the target data itself.

The overall accuracy achieved was 97.3% with a univariate distribution, the probability distribution of one random variable, of 99.7%, and a bivariate distribution, probability distribution of a random vector consisting of multiple random variables, of 97.3%. This confirms that Mostly AI is an effective platform and the generated synthetic data is representative of the raw data and can be depended upon in cases of law datasets

6.7 Evaluation of synthetic data Model Performance on test data

From the test set accuracy of 97.78% was achieved with the model based on the actual dataset with an accuracy of 93.33% being achieved on the model based on synthetic data when the same test data was used. Fig. 44 gives the confusion matrix test data classification based on the synthetic data-based model. The results prove that synthetic data can be relied upon for the training of models for the prediction of respiratory diseases.

ACCURACY
93.33%



	COPD_INFECTED	HEALTHY_PERSON	UNCERTAIN
COPD_INFECTED	100%	0%	0%
HEALTHY_PERSON	0%	85.7%	14.3%
F1 SCORE	1.00	0.92	

Fig. 44: Synthetic data model test result

6.8 Effect of data size on model Accuracy

The performance of the models was also evaluated based on the size of the dataset. Fig. 45 shows a plot of accuracy against the size of datasets. The results show that the accuracy increases as the data size increases up to until an adequate amount of data is reached. This proves the need to synthetically enhance data in cases of low datasets.

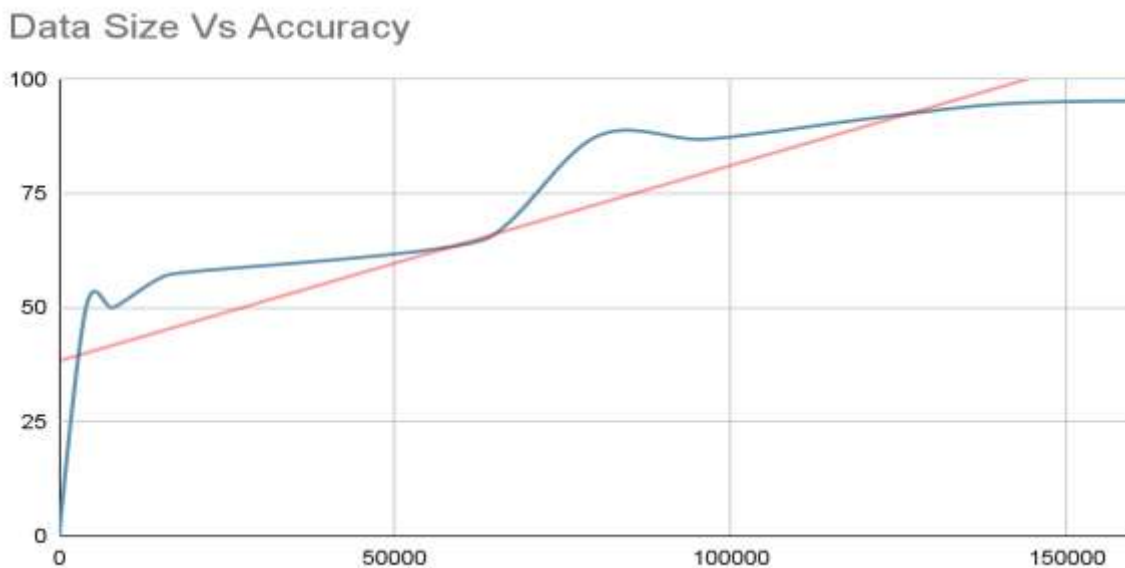


Fig. 45: A plot of data size vs accuracy of the model

CONCLUSION

Respiratory disease is one of the main causes of death across the globe. The use of IoT and AI provide opportunities that can be explored for early prediction of the diseases, However, the development of better ML models for the prediction of respiratory diseases has been hindered by lack of enough datasets and privacy issues limiting access to data. This research thesis presents an embedded experimentation-driven methodology for designing and developing portable edge AI-based disease prediction kits, focusing as a starting point to early prediction of COPD from exhaled breath signatures before scaling to other respiratory diseases. From experimentation, we successfully validated the edge AI inference accuracy to predict COPD to be very similar to the optimal accuracy when using the cloud for inferencing. We analyzed the impact of varying different embedded real-time resources such as the number of exhaled breath sensors on both inference accuracy and embedded processor memory requirements.

This study also explored the use of synthetic data as a solution to the lack of datasets. An evaluation was done with open data sets on COPD. Two models were trained one based on an open dataset and the other on synthetic data, with both models performing at almost the same accuracies of 96.9% and 94.2% respectively. This confirms the hypothesis that synthetic data could be used in cases of low datasets for better ML for the prediction of respiratory diseases. In addition, the study shows that TinyML can be used to train ML models for the prediction of respiratory diseases on an embedded device. The resulting model requires limited resources with the simulation result showing the possibilities for embedded inference. The implementation of the proposed solution will help overcome the problem of limited datasets in healthcare. This will lead to better ML models and thus less dependent on medical personnel and reduced costs. In addition, the prediction of respiratory diseases at the edge will ensure that the challenges of connectivity and privacy are addressed.

We conclude that precision edge AI should go beyond optimization of embedded processor specs alone as done by edge AI model generation frameworks to consider the joint co-design of the edge AI model (software) and the main embedded hardware components such as processor peripherals.

Future works will involve the collection of datasets from volunteers suffering from other respiratory diseases to predict more diseases. We will also explore synthetic data generation to produce artificial data for other respiratory diseases.

REFERENCES

- [1] “WHO EMRO | Respiratory tract diseases | health topics.” <http://www.emro.who.int/health-topics/respiratory-tract-diseases/index.html> (accessed Mar. 30, 2021).
- [2] “WHO EMRO | Chronic obstructive pulmonary disease (COPD) | health topics.” <http://www.emro.who.int/health-topics/chronic-obstructive-pulmonary-disease-copd/index.html> (accessed Mar. 30, 2021).
- [3] “Mortality Due to Chronic Respiratory Diseases.” <http://chartsbin.com/view/3754> (accessed Jul. 18, 2021).
- [4] S. K. Dhar, S. S. Bhunia, and N. Mukherjee, “Interference aware scheduling of sensors in IoT enabled health-care monitoring system,” in *Proceedings - 4th International Conference on Emerging Applications of Information Technology, EAIT 2014*, Feb. 2014, pp. 152–157, DOI: 10.1109/EAIT.2014.50.
- [5] G. Acampora, D. J. Cook, P. Rashidi, and A. V. Vasilakos, “A survey on ambient intelligence in healthcare,” *Proc. IEEE*, vol. 101, no. 12, pp. 2470–2494, 2013, DOI: 10.1109/JPROC.2013.2262913.
- [6] A. Kaplan *et al.*, “Artificial Intelligence/Machine Learning in Respiratory Medicine and Potential Role in Asthma and COPD Diagnosis,” *J. Allergy Clin. Immunol. Pract.*, 2021, DOI: 10.1016/j.jaip.2021.02.014.
- [7] C. C. Moor *et al.*, “Exhaled breath analysis by use of eNose technology: A novel diagnostic tool for interstitial lung disease,” *Eur. Respir. J.*, vol. 57, no. 1, 2021, DOI: 10.1183/13993003.02042-2020.
- [8] E. Mekov, M. Miravittles, and R. Petkov, “Artificial intelligence and machine learning in respiratory medicine,” *Expert Rev. Respir. Med.*, vol. 14, no. 6, pp. 559–564, Jun. 2020, DOI: 10.1080/17476348.2020.1743181.

- [9] “OP-JAMI200260 360..364 | Enhanced Reader.”
- [10] A. Tucker, Z. Wang, Y. Rotalinti, and P. Myles, “Generating high-fidelity synthetic patient data for assessing machine learning healthcare software,” *npj Digit. Med.*, vol. 3, no. 1, 2020, DOI: 10.1038/s41746-020-00353-9.
- [11] A. Tiele, A. Wicaksono, S. K. Ayyala, and J. A. Covington, “Development of a compact, iot-enabled electronic nose for breath analysis,” *Electron.*, vol. 9, no. 1, 2020, DOI: 10.3390/electronics9010084.
- [12] P. Makh, R. Mudhalwadkar, and A. Babtiwale, “Sensor System Development for Bronchitis Detection from Exhaled Breath,” *Proc. Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2018*, no. Iccict, pp. 1440–1442, 2018, DOI: 10.1109/ICICCT.2018.8473092.
- [13] “SpiroNose | Breathomix.” <https://www.breathomix.com/spironose-2/> (accessed Mar. 30, 2021).
- [14] E. Li, L. Zeng, Z. Zhou, and X. Chen, “Edge AI: On-Demand Accelerating Deep Neural Network Inference via Edge Computing,” *IEEE Trans. Wirel. Commun.*, vol. 19, no. 1, pp. 447–457, 2020, DOI: 10.1109/TWC.2019.2946140.
- [15] “Advancing Lung Health in Africa, a global imperative: The Pan African Thoracic Society (PATS) - ERS - European Respiratory Society.” <https://www.ersnet.org/news-and-features/news/advancing-lung-health-in-africa-a-global-imperative-the-pan-african-thoracic-society-pats/> (accessed Jul. 17, 2021).
- [16] “Over three million suffer from respiratory diseases annually – report | The New Times | Rwanda.” <https://www.newtimes.co.rw/news/over-three-million-suffer-respiratory-diseases-annually-report> (accessed Jul. 17, 2021).
- [17] “Connecting Africa - Internet: Africa starts to open its window to the world.” <https://interactive.aljazeera.com/aje/2016/connecting-africa-mobile-internet-solar/internet-connecting-africa.html> (accessed Jul. 18, 2021).
- [18] A. Smolinska, A. C. Hauschild, R. R. R. Fijten, J. W. Dallinga, J. Baumbach, and F. J. Van Schooten, “Current breathomics - A review on data pre-processing techniques and machine

- learning in metabolomics breath analysis,” *Journal of Breath Research*, vol. 8, no. 2. Institute of Physics Publishing, 2014, DOI: 10.1088/1752-7155/8/2/027105.
- [19] T.-C. Kuo *et al.*, “Human Breathomics Database,” *Database*, vol. 2020, Jan. 2020, DOI: 10.1093/database/baz139.
- [20] A. VM, “Microelectronic Gas sensors for Non-invasive Analysis of Exhaled Gases,” *J. Nanomed. Nanotechnol.*, vol. 11, no. 3, 2020, DOI: 10.35248/2157-7439.19.11.544.
- [21] “Breath Test for Hydrogen, Methane and Hydrogen Sulfide | Trio-smart.” <https://www.triosmartbreath.com/> (accessed Mar. 30, 2021).
- [22] “Smart Breathing Detection Integrated with Pulse Oximetry - Arduino Project Hub.” <https://create.arduino.cc/projecthub/ahmadradhy/smart-breathing-detection-integrated-with-pulse-oximetry-9a0f83> (accessed Jul. 18, 2021).
- [23] A. Velásquez, C. M. Durán, O. Gualdron, J. C. Rodríguez, and L. Manjarres, “Electronic nose to detect patients with COPD from exhaled breath,” *AIP Conf. Proc.*, vol. 1137, pp. 452–454, 2009, doi: 10.1063/1.3156579.
- [24] S. Dragonieri, G. Pennazza, P. Carratu, and O. Resta, “Electronic Nose Technology in Respiratory Diseases,” *Lung*, vol. 195, no. 2. Springer New York LLC, pp. 157–165, Apr. 01, 2017, DOI: 10.1007/s00408-017-9987-3.
- [25] A. Kononov *et al.*, “Online breath analysis using metal oxide semiconductor sensors (electronic nose) for diagnosis of lung cancer,” *J. Breath Res.*, vol. 14, no. 1, 2020, DOI: 10.1088/1752-7163/ab433d.
- [26] K. Moessner and E. Shakshuki, “Detection of Ketosis Using Non- Invasive Method,” vol. 01, no. 01, pp. 6–10, 2020.
- [27] C. M. Durán Acevedo, C. A. Cuastumal Vasquez, and J. K. Carrillo Gómez, “Electronic nose dataset for COPD detection from smokers and healthy people through exhaled breath analysis,” *Data Br.*, vol. 35, pp. 4–9, 2021, DOI: 10.1016/j.dib.2021.106767.
- [28] M. Boubin and S. Shrestha, “Microcontroller Implementation of Support Vector Machine

- for Detecting Blood Glucose Levels Using Breath Volatile Organic Compounds,” *Sensors (Basel)*, vol. 19, no. 10, 2019, DOI: 10.3390/s19102283.
- [29] A. V. Radogna, P. A. Siciliano, S. Sabina, E. Sabato, and S. Capone, “A low-cost breath analyzer module in domiciliary non-invasive mechanical ventilation for remote copd patient monitoring,” *Sensors (Switzerland)*, vol. 20, no. 3, 2020, DOI: 10.3390/s20030653.
- [30] S. Mansouri, S. Boulares, and T. Alhadidi, “Non-invasive Measurement of Blood Glucose by Breath Analysis,” *IEEJ Trans. Electr. Electron. Eng.*, vol. 15, no. 10, pp. 1457–1464, 2020, DOI: 10.1002/tee.23216.
- [31] O. Lawal, W. M. Ahmed, T. M. E. Nijsen, · Royston Goodacre, · Stephen, and J. Fowler, “Exhaled breath analysis: a review of ‘breath-taking’ methods for off-line analysis,” *Metabolomics*, vol. 13, p. 110, 123AD, DOI: 10.1007/s11306-017-1241-8.
- [32] “Smart Breathing Detection Integrated with Pulse Oximetry - Arduino Project Hub.” <https://create.arduino.cc/projecthub/ahmadradhy/smart-breathing-detection-integrated-with-pulse-oximetry-9a0f83> (accessed Mar. 30, 2021).
- [33] T. Saidi *et al.*, “Non-invasive prediction of lung cancer histological types through exhaled breath analysis by UV-irradiated electronic nose and GC/QTOF/MS,” *Sensors Actuators, B Chem.*, vol. 311, 2020, DOI: 10.1016/j.snb.2020.127932.
- [34] “BreathBase® Data | Breathomix.” <https://www.breathomix.com/breathbase-data/#> (accessed Apr. 04, 2021).
- [35] “Breath Biopsy Services quality breath biomarker analysis.” <https://www.owlstonemedical.com/services/> (accessed Jul. 18, 2021).
- [36] “Lablicate.” <https://lablicate.com/platform/openchrom> (accessed Jul. 18, 2021).
- [37] P. Weber, J. K. Pauling, M. List, and J. Baumbach, “Balsam—an interactive online platform for breath analysis, visualization and classification,” *Metabolites*, vol. 10, no. 10, pp. 1–16, Oct. 2020, DOI: 10.3390/METABO10100393.
- [38] L. Greco, G. Percannella, P. Ritrovato, F. Tortorella, and M. Vento, “Trends in IoT based

solutions for health care: Moving AI to the edge,” *Pattern Recognit. Lett.*, vol. 135, pp. 346–353, Jul. 2020, DOI: 10.1016/J.PATREC.2020.05.016.

- [39] “Home - AI-generated Synthetic Data.” <https://mostly.ai/> (accessed Jul. 18, 2021).
- [40] P. Sethi and S. R. Sarangi, “Internet of Things: Architectures, Protocols, and Applications,” *J. Electr. Comput. Eng.*, vol. 2017, 2017, DOI: 10.1155/2017/9324035.
- [41] “Nano 33 BLE Sense | Arduino Documentation | Arduino Documentation.” <https://docs.arduino.cc/hardware/nano-33-ble-sense> (accessed Jul. 29, 2021).
- [42] “MQ-3 Alcohol Sensor Module Pinout, Datasheet & Other Gas Sensors.” <https://components101.com/sensors/mq-3-alcohol-gas-sensor> (accessed Aug. 01, 2021).
- [43] “MQ-135 Gas Sensor Pinout, Features, Alternatives, Datasheet & Uses Guide.” <https://components101.com/sensors/mq135-gas-sensor-for-air-quality> (accessed Aug. 01, 2021).
- [44] “MQ138 VOC Gas Sensor--Winsen.” <https://www.winsen-sensor.com/sensors/voc-sensor/mq138.html> (accessed Aug. 01, 2021).
- [45] “16x2 LCD Pinout Diagram | Interfacing 16x2 LCD with Arduino.” <https://www.electronicsforu.com/technology-trends/learn-electronics/16x2-lcd-pinout-diagram> (accessed Aug. 01, 2021).

APPENDICES

Appendix 1: Paper Acceptance notifications

Your paper #1570749726 ('Edge AI-Based Respiratory Disease Recognition From Exhaled Breath Signatures')   

JEEIT 2021 jeeit2021-chairs@ieee.org
to: Mr. Dalaleneh, Jean, Jimmy, Jeeit, Khalid, Iyad, Ali, Muhammad +
Sep 18, 2021, 4:33 PM (12 days ago)   

Dear Mr. Samson Ooko

Congratulations! On behalf of the Conference Committee of the 2021 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), we are happy to inform you that your paper #1570749726 entitled:

(Edge AI-Based Respiratory Disease Recognition From Exhaled Breath Signatures)




has been accepted for presentation and inclusion in the Proceedings of JEEIT 2021, published by IEEE.

We are proud to inform you that JEEIT 2021 has received a large number of excellent submissions. Each submission was reviewed by several experts in the field and the committee chose a subset of those best submissions based on the reviews.

Please see the reviewers' (on your paper comments below. These reviews can also be found at <https://ieeexplore.ieee.org/abstract/document/9511570749726>). These comments are intended to help you to improve your paper for final publication. The review comments should be addressed, as final acceptance is conditional upon appropriate response to the requirements and comments. The conference committee retains a list of certain critical comments to be addressed by authors, and will confirm that these have been addressed in the camera-ready version.

JEEIT 2021 implements IEEE's conference no show policy. JEEIT 2021 policy is to exclude a paper from distribution after the conference (e.g., removal from IEEE Xplore) if the paper is not presented at the conference. However, authors from outside Jordan who can not attend the conference due to Covid-19 pandemic, will be allowed to present their papers virtually and have reduced registration fees (25% discount from the original fees).

[GC 2021 Workshop - SESSy] Your paper #1570743875 ("TinyML in Africa: Opportunities and Challenges")   

GC 2021 Workshop - SESSy gc2021-workshop-essy-chairs@ieee.org
to: Mr. Marvin, Jimmy, Marco, Ruben, Patric, Natalia, Coco +
Thu, Sep 16, 9:29 AM (11 days ago)   

Dear Mr. Samson Ooko:

On behalf of the Workshop Program Committee, we are pleased to inform you that your paper #1570743875 entitled "TinyML in Africa: Opportunities and Challenges" has been accepted for presentation at the IEEE GLOBECOM 2021, December 7 - 11, in Madrid, Spain, and virtual, for publication in its proceedings. Please revise your paper carefully to address the reviewers' comments and suggestions, and to ensure that your final paper fits the camera-ready format. The reviews are available at <https://ieeexplore.ieee.org/abstract/document/9511570743875>.

FORMAT AND SESSION
Please note that by submitting manuscripts, we assume that the authors have agreed to present their accepted papers at sessions organized by the Workshop Chairs.

COVID-19 UPDATE
The IEEE GLOBECOM 2021 organizing team is closely monitoring the development of the COVID-19 pandemic and the government policy in response to it. In case of travel restrictions that prohibit physical appearance, virtual presentation will be accepted as an alternative of in-person presentation. The exact hybrid (in-person + virtual) format of GLOBECOM 2021 will be announced soon on the GLOBECOM 2021 website <https://globecom2021.ieee-globecom.org/> and via email.

[I3CAT'21] Your paper #1570745657 ('Synthetic Exhaled Breath Data Based Edge AI Model for Prediction of Chronic Obstructive Pulmonary Disease (COPD)')   

I3cat21-chairs@edas.info
to: Mr. Dalaleneh, Jean, Jimmy +
Mon, Aug 16, 2:28 PM   

Dear Mr. Samson Ooko:

Congratulations - your paper #1570745657 ('Synthetic Exhaled Breath Data Based Edge AI Model for Prediction of Chronic Obstructive Pulmonary Disease (COPD)') for I3CAT'21 has been **accepted** and will be presented at the session titled: __

The reviews are below or can be found at <https://ieeexplore.ieee.org/abstract/document/9511570745657>.

Appendix 2: Publication Peer Reviews

Review 1

Significance	Originality	Relevance	Presentation	Formatting	Abstract & Introduction
Very significant (5)	Very Original (5)	Very Relevant (5)	Very good (5)	Excellent (5)	Very adequate (5)

Technical Quality	Experimentation	Conclusions	References	Overall rating
Very good (5)	(4)	Very clear (5)	Very appropriate and up-to-date (5)	Very good (5)

Review 2

Significance	Originality	Relevance	Presentation	Formatting	Abstract & Introduction
Very significant (5)	(4)	Very Relevant (5)	Very good (5)	(4)	(3)

Technical Quality	Experimentation	Conclusions	References	Overall rating
Very good (5)	(4)	(4)	Very appropriate and up-to-date (5)	Very good (5)

Review 3

Presentation	Organization	Originality	Relevance	Acceptance Score
Very Good (4)	Very Good (4)	An Interesting contribution (3)	Very relevant (4)	Totally Accepted (4)

