An IoT-Based Ambient Assisted Living for Elderly Care and Monitoring in COVID-19 Pandemic Using Artificial Intelligence and Deep Learning

by

Shovon Bhowmick
17101208
TAREK FERDOUS
17101491
RAIHAN MOMTAZ
17101196

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering
Brac University
June 2021

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Shovon Bhowmick
17101208

TAREK FERDOUS
17101491

RAIHAN MOMTAZ
17101196

Approval

The thesis titled "An IoT-Based Ambient Assisted Living for Elderly Care and Monitoring in COVID-19 Pandemic Using Artificial Intelligence and Deep Learning" submitted by

1. Shovon Bhowmick (17101208)
2. TAREK FERDOUS (17101491)
3. RAIHAN MOMTAZ (17101196)

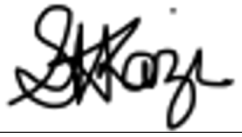
Of Spring, 2021 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on June, 2021.

Examining Committee:

Supervisor:
(Member)

Dr. Md. Golam Rabiul Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)



Sadia Hamid Kazi
Chairperson and Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

In late 2019, a novel Coronavirus broke out from China, which has dispersed all over the globe and has taken away countless lives. Despite the fact that every person is at risk of getting infected with the virus, older people are more likely to fall victim to the virus due to their declining immune systems. Although there has been significant development of vaccines, it is seen that the mutation of the COVID-19 has made it tough to control with the medication available. Due to an uncountable number of Coronavirus strains, many countries are now facing the second wave of the pandemic. This disease is very contagious. Assisted living technologies are evolving with time to give people a better life. This technology can be used for older people in Coronavirus pandemic situations. Most of them have physical and cognitive impairments and face immense challenges in their day-to-day life. Older people are vulnerable to disease, and even simple disease can worsen their health. If our older people stay healthy and safe, our world would be a better place. In this paper, we have proposed an IoT-architectured system incorporated with Artificial intelligence and deep learning that can help diagnose a disease of our beloved aged people. The proposed architecture will collect all the data from different medical IoT sensors and relay them to the cloud, where the system will process and help us monitor the health of older people. This information will be seen from a dedicated dashboard. The system will be able to predict the possible COVID-19 disease that an elderly person may suffer in the near future. We cannot imagine a world without our dearest older people, and the chances of the next pandemic cannot be eliminated either. In order to be prepared for any future pandemic, this type of system will be beneficial.

Keywords: COVID-19; Deep Learning; Artificial Intelligence; Sensor; Prediction; Monitoring; Linear Regression Analysis; Hidden Markov Model; XGBoost; SVM; ESP32; Raspberry Pi

Dedication

This dissertation is dedicated towards all the front-line workers who are risking their life for the betterment of the society.

Acknowledgement

First of all, all praise to Almighty for whom we are able to complete the whole thesis

Secondly, to our supervisor, Dr. Md. Golam Rabiul Alam for guiding us through the whole process. Without his directions and motivation this thesis was near impossible.

Finally, to our parents for supporting us and believing us throughout the whole life

Table of Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AAL Ambient Assisted Living

HMM Hidden Markov Model

IoT Internet of Things

KIT Keep in Touch

ML Machine Learning

NFC Near Field Communication

PIR Passive Infrared

RBF Radial Basis Function

RFID Radio-Frequency Identification

SVM Support Vector Machine

Chapter 1

Introduction

1.1 Ambient assisted living

The ratio of older people to younger people is increasing dramatically, especially in the developing world [1]. So, family members have a hard time dealing with the healthcare costs, and in some cases, there is no one to look after the older people or pay for their healthcare. Ambient Assisted Living is a branch of ambient intelligence, which uses ambient intelligence techniques for exchanging communication between older people and the environment in which he/she is occupying [2]. This technology uses different sensors and connecting mediums to help older people live a normal life. The exploration is mainly done on their movement, vitals, and behaviors [1].

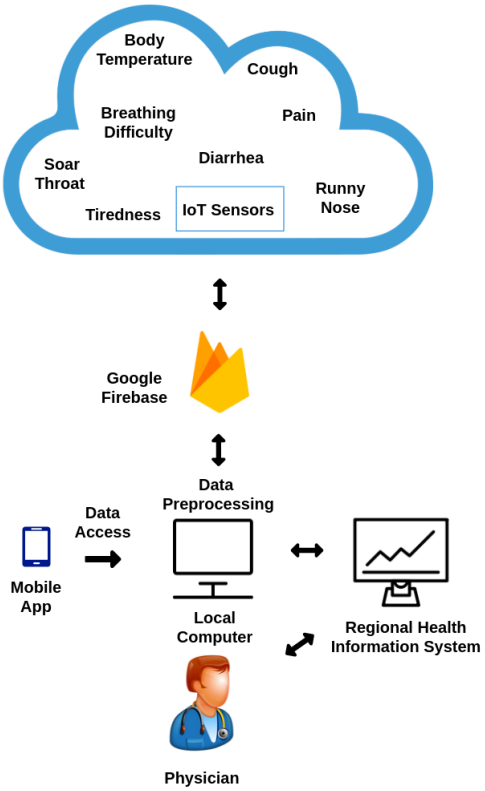


Figure 1.1: Ambient Assisted Living

Ambient intelligence is a unit of pervasive computing, which gathers the user's data by wireless communication smart, intelligent sensors network and built-in microprocessor in those sensors, without disturbing the normal day-to-day activities of the people [3]. These sensors are very receptive to the changes in the environments and are almost invisible to the users. For example, fall detection can be done by computer vision, health analytics can be used to make Machine Learning models and use for future detection of diseases, etc.

1.2 Internet of Things

IoT is the interconnection of various sensors/devices through the internet. The sensors can communicate by the wireless sensor networks and exchange the information they collect over a particular time without any human assistance. These sensor data are sent to the cloud for processing. The results can then be viewed over a website or mobile app [4]. In this way, we can monitor and keep track of anyone or any place from babies to adults. These sensors are affordable and consume less power, and with the help of the internet and the availability of free access to cloud platforms such as firebase, thingspeak etc has open the door for the researchers to explore a lot of areas.

1.3 Worldwide COVID-19 situation

Coronavirus, a virus infecting disease, has grasped the world in this early era of the 21st century. Millions of people are infected by it, and nearly a million people have died till now (case: 150.7m and death: over 3m) [5]. Only a few diseases like this were spread across borders this vastly. For this, WHO (World Health Organization) has declared this contagious disease to be a pandemic [6]. Coronavirus devastates the respiratory system of a patient, and in severe cases, the patient dies [7]. The structure of the virus has many variations [8]; as a result, scientists were not capable of finding a medicine or a proper vaccine that gives 100% protection against all the strains of coronavirus till now which deteriorates the situation even more. However, younger people with good immune systems are less likely to be infected by it [9], although they may be carriers of the virus themselves. The rate of infection is much higher for older people. They can restrict themselves from going outside and get in contact with other people, but most of their family may have younger people who go outside for groceries or earning livings etc. As they grew old their health has weakened. They get affected by any simple disease easily. We can see statistics of death counts by Coronavirus, and a major part of it is our beloved old-timers.

1.4 Overview

We all know how older people struggled through their lives at their younger age. The hardship they have gone through is what keeps us where we are now. We cannot be more grateful towards them. Now in the autumn of their life, we cannot risk them. So, we are implementing an AAL environment (Ambient Assisted Living Environment) using an IoT structured system for monitoring older people. Ambient assisted living falls under the category of ambient intelligence where elder people are

benefitted and live more independently [2]. The main objective of AAL is to comfort older people and physically challenged people. Also, in the present era, we started to acknowledge the usefulness of IoT (Internet of Things), although the concept of a network of the smart network was first discussed in earlier 1982. IoT devices connected to a network improves analysis, planning as we get real-time data through some always-on IoT sensors or IoT devices. We can change the present infrastructure drastically with an IoT system. If we implement this system with elder health care, we can minimize our aged people getting affected by this contagious Coronavirus. The system will monitor older people and analyze symptoms of Coronavirus through IoT devices. Then if an anomaly is found, the system will alert. Also, as we are monitoring older people in real-time, we can also provide care so that they can lead healthier life. Their overall comfort is our goal.

1.5 Motivation

The COVID-19 is a very contagious disease. In the assisted living homes or environments, a lot of older people live together. If any one of the people who are living in the assisted environment contacts the disease, there are chances that most of them will eventually get it. There are "Ambient Assisted Living" systems which monitors older people's data but are not tailored properly for COVID-19. There are no well-known "Ambient Assisted Living" systems which actively monitor the symptoms and help to separate the people who might be at risk to contact the disease already. If we depend on the caretakers for collecting personal reports of the older people for diagnosing COVID-19, then there are chances that the older patient might be already in their final stage, and on top of that, they might already have passed the virus to the surrounding, creating a deadly scenario for everyone. Older people are the most vulnerable group for this disease but their survival rate can improve if the diagnosis is done faster and they are treated in their initial stages. Our prescribed "Ambient Assisted Living" system will help to diagnose the disease as fast as possible based on the real-time sensor readings and Machine Learning algorithm which will increase the survival rate of the older person contacting the disease. Moreover, we can help other people not to contact the disease by isolating the possible carriers.

1.6 Problem Statement

With time human beings slowly lose vigor and immunity. As people get old, they become weak, their health condition deteriorates over time. Their necessary organs start to malfunction and as a result, they fall prey to numerous diseases. As they get financially dependent on their children after their retirement they become very lonely. They miss their workplace and really become sad. For this reason, a lot of them decide to live in the "Assisted Living Facility". In the "Assisted Living Facility," a lot of old people share the same space. A lot of research has been done to incorporate the technology in the "Assisted Living" environment. Since COVID-19 is uncontrollable, new variants of the virus are spreading rapidly, old people are the easy targets for the virus and since the body is unable to fight with its existing capabilities they get overdone. According to the American Association of Retired Persons (AARP), of all the deaths in the USA, 95 percent have been older people

[10]. Many people who are affected by COVID-19 are not sure which healthcare to go to get tested as the healthcare has a likely chance to spread the virus and prioritizing them by calling the infected to get to know their current condition [11] is tough and irritating, this problem can be avoided if the patients and healthcare workers are monitored and data are continuously sent to the hospital and processed to avoid the spread from the healthcare system. COVID-19 can be diagnosed with the help of key symptoms and machine learning models trained on the symptoms dataset [12]. Although this research [12] helped us to detect COVID-19 with the help of Machine Learning, they did not propose any model to collect the data of the symptoms in real-time. With the advent of Linux-based light operating systems, the IT market is filled with different wearables. These wearables can be smart-watches, smart-bands, etc. Most people like to keep track of their health with these smart-watches [13]. The biggest problems with these wearables are their API is not good enough for sharing the data for public research and live monitoring, they do not have all the sensors built-in for our framework, they are not programmed to send all the sensor data information concurrently. To resolve this issue we are proposing a framework that will be tailored particularly for monitoring COVID-19 symptoms. This will use the multi-threading capability of the modern CPUs to send all the data concurrently without missing any key moment. We will also make Machine Learning models which will make a prediction whether the person is a possible COVID-19 case by comparing with the trained models.

1.7 Research Objectives

The research goals that we want to accomplish are: (1) An IoT based framework for assisted living facility and (2) Prediction of COVID-19 disease of the people living in the facility

1. An IoT-based framework to collect and monitor data which are related to COVID-19 symptoms in "Ambient Assisted Living Facility" and detect whether a person is carrying COVID-19. In order to make the framework, we have to select a temperature sensor, blood oximeter sensor, occupancy detection sensor, tissue counter sensor, etc. and collect data without interrupting elderly people. The details of the proposed model is discussed in the Methodology section. This framework will facilitate collecting data from the people which are related to COVID-19 symptoms in real-time.

2. Individually verifying and predicting the sensor values by Hidden Markov Models. Moreover, using Logistic Regression, Decision Tree, Random Forest, XGBoost, SVM and Basic Neural Network for selection of the most suitable Machine Learning/Deep Learning algorithms by comparing them. Then build an automated system to send the processed sensor data to the trained model and based on the best performing Machine Learning/Deep Learning algorithm prediction, inform the authority if any person is a suspect of COVID-19 disease.

Chapter 2

Literature Review

The main idea of implementing a health-based IoT system that will predict Coronavirus infection in older people was inspired by the paper we found titled as "The Internet of Things for Ambient Assisted Living" [14]. Here the authors proposed IoT models for many aspects, and health was one of them. According to them, AAL (Ambient Assisted Living) supports older people, and the primary goal of AAL is to benefit individuals, the economy, and society. Here, they have proposed two IoT models. One is KIT (Keep in Touch). Here, IoT sensors with RFID are used to collect data, and through mobile phones, the data is shown to the user. The touch device is actually a smartcard containing the NFC wireless chip and other measuring sensors. To measure anything, the user has to touch the card and tap the card on their mobile phone. However, the model does not have any processing and feedback, and the process requires a mobile phone to update data which delays the result. Another model is "Closed Loop Healthcare Services"; here, the new feature is a simple analysis and giving feedback to the user. For analyzing, different types of algorithms are used. Both models were famous at that time but the models were very simple and did not use any neural network or deep network for predicting. So, we needed more updated works.

We found another paper focusing on AAL for elder people [15]. Here, according to the author, they tried to implement an architecture for ambient assisted living that supports pre-hospital urgency and remote monitoring of patients with severe conditions. They have cited that 600 million people are living currently in the world over the age of 60, and the number will get doubled by 2025. The model they have proposed has many components. Among them some are- Home automation, Security, Ambient Intelligence, and Telemedicine unit. The model worked in all those areas so that the elderly people can have a more comfortable and independent life. The model is pretty simple but shows the greatness of IoT.

In this paper [16], the authors (Otoom M, Otoum N, Alzubaidi MA, Etoom Y, Banihani R, 2020) presented a system that has the ability to predict and monitor in advance the Coronavirus cases with the help of health informatics. The sensors used to collect the symptom data are all wearable sensors. The author describes the framework such that the data obtained from the sensors is passed to the data

analyzing center with the help of the cloud. The data communication continues between healthcare and the cloud. The data analysis center receives the data from the health care center, and as it has the ability to host algorithms, it can continuously update its models. Then by the use of symptom data collected from the sensors the model can tell whether or not a person is infected or not in real-time based on its previous and past data. The results are displayed in the dashboard. These results of the infected are extremely important for the physician as they can have more knowledge and understanding of the virus. The patient is then assigned to a physician and would be called by the healthcare center to perform the Polymerase Chain Reaction (PCR), which is used to check whether a person is COVID-19 positive or negative. If positive, then the patient is sent to isolation centers for treatment. For this paper, the author has used eight algorithms. After the result was obtained, the author concluded that out of the eight, five algorithms produce an accuracy of over 90 percent. Thus, the author concluded that COVID-19 could be detected flawlessly with this process.

The authors (M. McMichael, et al., 2020) in paper [17] did surveillance on the people of King County, Washington, by contacting the affected and likely to get affected people directly over the telephone. They collected data regarding their symptoms, their current well-being or condition and if they have been in contact with anyone who has been Covid- positive, etc. They got this information from the people who had visited the Health Care and processed this information to prioritize the Health Care which is safe to get tested or get emergency care for the affected, as people in Healthcare have chances of spreading the virus. This risk of transfer of Covid-19 could have been avoided if they could monitor the affected and the healthcare givers continuously as we used in our proposed framework. Moreover, in the paper, it is mentioned they were calling everyone; most of the time, people who are sick would not like to talk so much. And if the caregivers were monitored, they would also get a warning and take necessary precautions.

In this paper [18], the authors (Taiko O., Ezugwu A.E, 2020), proposed an intelligent home healthcare facility for COVID detection, that would make the lives of elderly people easier, the author made the user use a mobile phone to control all the home appliances present in the environment from the room temperature to the fans, lights television, etc. The author has used multiple sensors to collect the user's data regarding body temperature, pressure, blood pressure, etc. In addition to the IoT environment, the author has also included a caregiver or a second body, who would measure all the physical conditions using the measuring instrument we find in healthcare these data obtained from the mobile application would be sent to the doctors. As a result, there are both manual and automated systems incorporated in this research and the system is more likely to perform better than if only one was present. The author is also sending these data to the healthcare and if the user has a bad condition they can easily contact doctors live and take necessary medication, the mobile application also notifies the user about when to take which medicine.

In the paper [19], the authors (Achirei, S. D. et al.,2020) proposed a platform where different sensors and devices can be connected in order to make them work autonomously. This is both a software and hardware solution. If the wireless technology is used then proper care must be taken to ensure every component is getting full access to the network. The authors proposed a layered format such that any device or sensor can be changed without thinking about compatibility. In order to get real-time data CISC architecture CPU should be used. The authors also suggested all the sensors/devices should be designed with the driver inside them such that with API the sensor/devices can be easily added or disconnected.

In the paper [20], the authors (Taleka and S, 2019) reviewed different Neural Network algorithms to monitor human activity. They emphasized on building an ecosystem that correlates well with the technological domains such as Pervasive Computing, Sensor networks, etc. Firstly, to get the information about the vital signs, the possible sensor names were discussed. Wearable/body sensors, ambient sensors, and hybrid sensors are the sensors normally used in these types of scenarios. Then they discussed the deep learning architectures used in ambient assisted living. Different versions of Discriminative Deep Architecture, Generative Deep Architecture, and hybrid of the aforementioned were used to perceive different activities by them. They then discussed why video-based supervision is inaccurate and why sensor-based HAR is the way to go in order to apply deep learning techniques. They also reviewed the recognition performance using deep learning architectures based on sensor modality. After the comparison, the authors said we should monitor fever, heart state, and breathing for vitals. The authors concluded by mentioning how deep learning can help to reduce the cost to make an ambient assisted living system.

The authors (J. S. Obeid et al., 2020) in paper [21] tried to use deep learning techniques to improve the screening mechanism for COVID-19 testing. They found out the greatest number of represented words in the self-reported documents after the usual pre-processing of the data. They discussed how they screened the patients showing symptoms and decided which patients should be tested with RT-PCR because the number of test kits was small compared to the population that is showing symptoms. We think this method can be used in ambient assisted living because the self-reports of the members of the nursing home can be analyzed with this algorithm. They mentioned that besides the state-of-the-art NLP methods, ML-based classifiers helped them cluster related medical data. The authors used CNN deep learning architecture with the input layer dimension of 628, filter sizes of 3,4, and 5, and ReLu activation function. The authors found some interesting things out of the result, patients' self-assessment reports, which contained the words like "smell," "taste," "sense," etc., were more prone to get positive results. The authors reported that their model was accurate, although they were a little skeptical because the training data were a little bit small. They planned to improve the model by adding correlation with pre-existing risk factors and adding epoch to the training set. The authors concluded by emphasizing how these models can prioritize to select more vulnerable patients.

In order to eliminate the risk of contracting COVID-19 during medical assessment the authors (Ruizhong et al., 2020) in [22] tried to assess the utility of 5G-based robot-assisted remote ultrasound systems. They divided the patient group into two parts and tried to diagnose COVID-19 by checking the heart and lungs via a remote system. The authors then discussed the harmful outcome of the current methods of assessments and using them in portable use cases. They offered ultrasound a possible replacement due to its several features like less harmfulness, portability, etc. But one of the downsides is, this needs close contact with the patients. They used MGIUS-R3, a robotic ultrasound system, to do the test, which resulted in quite a close result with that of HRCT without the drawbacks. They also did a retrospective study which confirms that the less severe cases were also determined. We think that this method is quite a fit for our project because this will eliminate some sort of chances of transmission of COVID-19 and due to its small form factor, it will fit in the "Ambient Assisted Living" echo system.

In this paper [23] the authors (Suryadevara N. K., Mukhopadhyay S.C.2014) developed a system that keeps track of the day-to-day activities done by an elderly citizen, this system measures how well an elderly citizen is. For the collection of data, a wireless sensor network (WSN) along with a home monitoring system software (HMSs) is used. For supervision, the pervasive sensor network (PSN) is used to evaluate the wellness of the object. The author can identify the activities of the object based on the Up-To-Date sent by sensors. The data obtained from the sensors is sent through the OpenVPN internet connection, and it is stored in a centralized SQL database. For data processing. For this system, the monitoring is done by visually displaying the wellness assessment on the website. The authors mainly used two functions known as "wellness function". The first wellness function computes the index level for inactive household objects this would give the healthcare provider an idea of whether the object is performing his/her daily activities normally. The second wellness function determines whether the object is doing any extra work more than that is done normally. In order to reduce the number of faulty messages, the author limited in a way such that identifying the objects location while supporting the wellness indices. In the results obtained from the system, the author concluded that this proposed model would provide sufficient evidence for the development and monitoring of the daily activities which can be obtained from the sensors.

After analyzing all these papers we have found some areas where we can do research in order to build a better framework for "Ambient Assisted Living" in a pandemic situation. Currently, there are no implemented hardware sensor frameworks to collect real-time data based on coronavirus symptoms. At present, the common practice to collect data for COVID-19 diagnosis is manual, either by calling them or responding through google forms but our research framework idea is to collect data automatically without any hindrance to the users. Moreover, our research will involve selecting the key symptoms for diagnosing the disease because there is no universal list of symptoms for diagnosis but in order to make the system more efficient, we have to select a finite number of symptoms and collect data based on that. Finally, we want to work on validating the values coming from the sensor by using machine learning to increase the accuracy of the diagnosis.

Chapter 3

Methodology

This section comprises of two extensive sections. In section 3.1 we have proposed an IoT-based framework for assisted living facility and section 3.2 will be about methods of prediction of the COVID-19 from the sensor data.

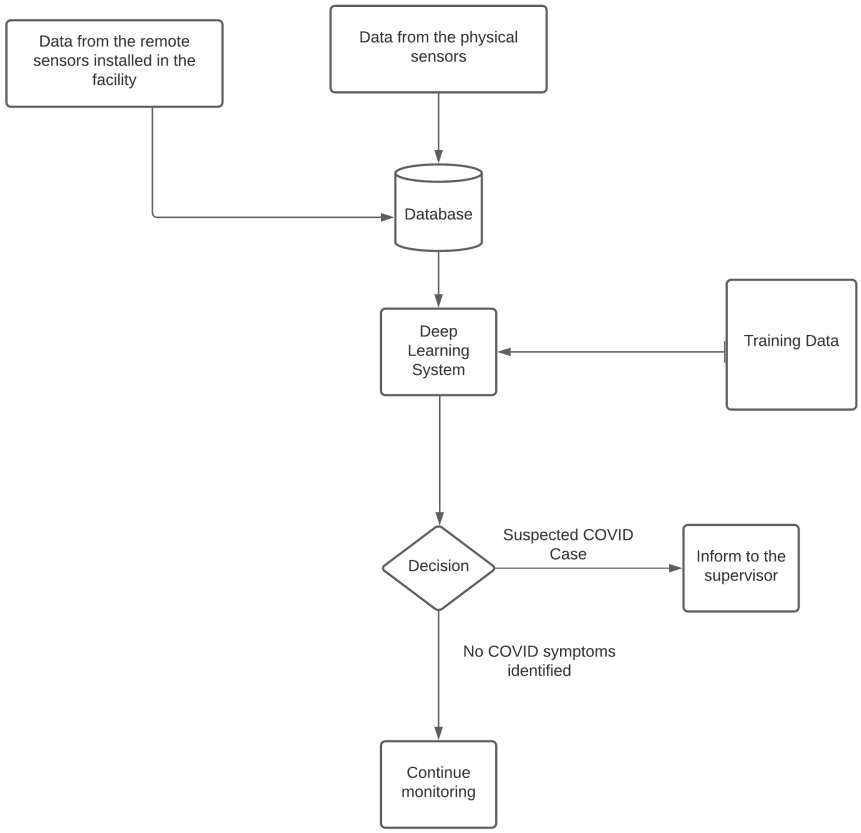


Figure 3.1: Workflow

The sensor will comprise of a temperature sensor, heart rate and blood oximeter, microphone, IR sensor, PIR sensor. We will set up all the sensors in the body and

also in the whole living environment. All sensors connected to the body will be non-invasive. Firstly, the deep learning system will be trained with the existing data and tested to set up the initial system. Then continuous sensor data will be sent to the system, which will be kept in the database and sequentially fed to the deep learning model for the decision purpose. If the system detects any anomaly, it will message immediately to the supervisor and this will help to separate them to stop further transmission and quick start of the treatment.

3.1 IoT framework for an ambient assisted living

The sensors are placed in mainly two places. For collecting the vitals data, non-invasive sensors like heart-rate and blood oximeter sensor, temperature sensor, microphone, switch are placed on the body. In order to detect diarrhoea, runny nose and occupancy, different sensors are placed in the facility. Our framework proposes a wearable watch having all the sensors required to collect the vitals. For research and development purposes, we have used Raspberry Pi 4 for body sensors and ESP32 for the remote sensors.

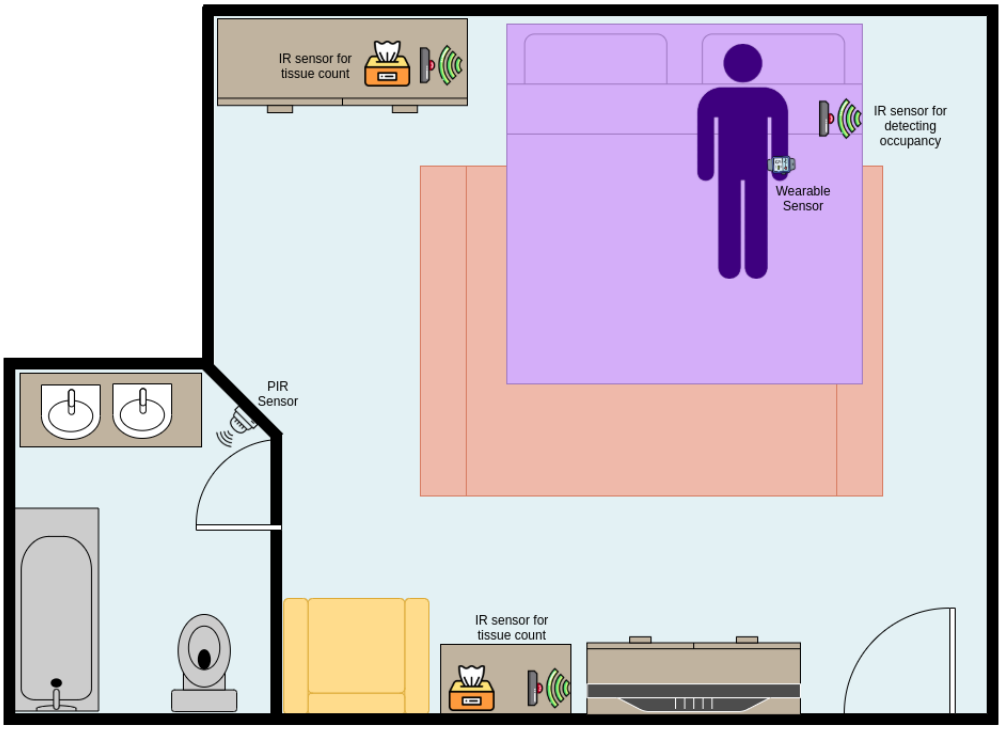


Figure 3.2: Floor Plan of the Sensors

3.1.1 Framework for collecting the body vitals

After researching the major symptoms of the COVID-19 disease we have come up with a list of symptoms that are common in COVID-19 patients [24] and can be easily detected by a number of sensors. These sensors can be easily assembled together to make a smartwatch. This system will have the following functions.

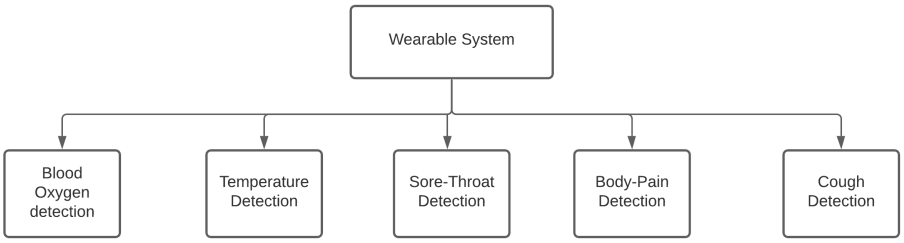


Figure 3.3: Symptoms Detection Using Wearable System

For implementing the prototype we have used Raspberry Pi. The sensors we have used are a temperature sensor for measuring the temperature of the body, a microphone for detecting the cough, a blood oximeter for detecting breathing difficulty, buttons for self-reporting of sore throat, and pain will be connected to the Raspberry Pi. Then Raspberry Pi will send these real-time data to the cloud.

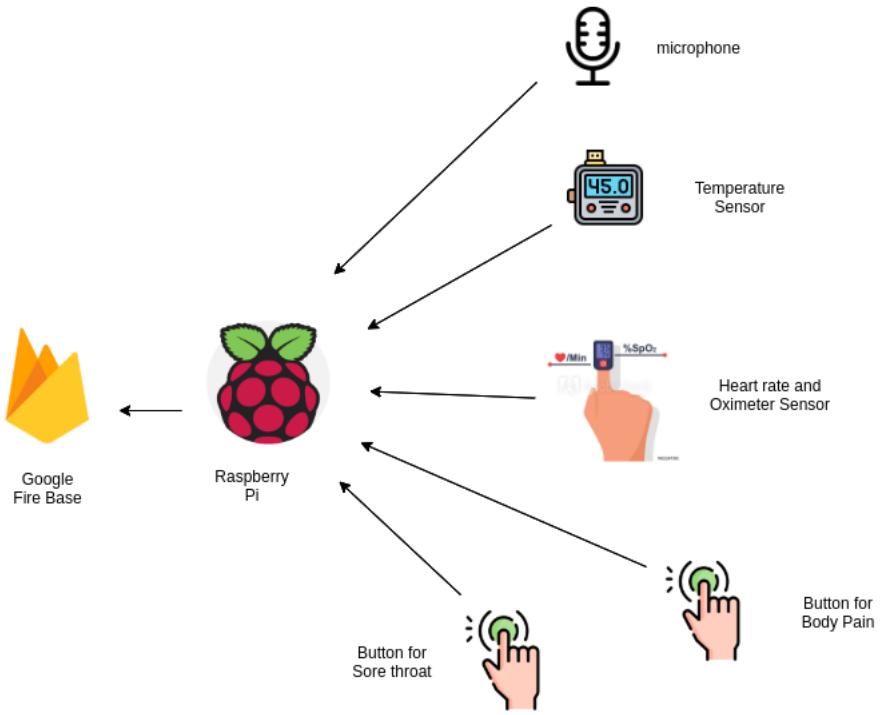


Figure 3.4: Proposed wearable system design

3.1.1.1 System for detecting the body temperature

In this pandemic situation, one of the primary symptoms is fever, so in order to constantly detect fever if we use a traditional method of measuring the fever then it will not be real-time. The traditional method requires a second person to read the value and input it. For this reason, we have decided to use a non-contact infra-red sensor [25], GY-906 MLX90614 which can be used to get the body temperature. This sensor is very robust and can be used to detect the temperature change, and whenever the temperature is higher than normal, it can be programmed in a way such that a warning could be given. This sensor is simple and helps in the detection of fever and helps to separate the infected [26]. Another reason to select this sensor is that it is small in size, which will be very easy to assemble in the wearable device.

This sensor communicates with the Raspberry Pi using the I2C protocol. The address of the I2C module is 0x5a. We can get both the atmospheric temperature value and object temperature value from the sensor, the object value is used to get the body temperature of the person.

Figure 3.5: GY-906 Infrared Temperature Sensor

Algorithm 1 Sensing the temperature and sending the value

```
1: procedure TEMPERATURE(sensor)                    ▷ temperature in celcius
2:     Thread Initialization
3:     I2C communication Initialization
4:     while sensor ==True do                       ▷ Keeping the system running
5:         Get the temperature value from the I2C module
6:         Send temperature value to the firebase
```

3.1.1.2 System for detecting the breathing difficulty

People who have been affected by the coronavirus, experience serious breathing difficulties. Oxygen is one of the vital elements needed by the body, research has shown that people who are affected by flu in a pandemic situation generally have low oxygen levels [27]. Oxygen level is one of the vital measurement factors in order to distinguish a healthy person from a sick person. The cells in the body aren't able to perform their usual function if there isn't sufficient oxygen available and if it occurs continuously they expire. As a result, organs also start to malfunction and cause serious problems to the body [28]. So in order to measure the oxygen level to detect the breathing difficulty, we have used MAX30100 [28][29], this sensor can be placed anywhere in the body because it is using reflectance oxygen saturation detection, The reflected light emitted from whichever body surface it is connected is

received by the photodiode and based on the intensity the oxygen level is measured [28]. This sensor communicates with the Raspberry Pi using the I2C protocol. The address of the I2C module is 0x57. We can get both the heart rate and oxygen saturation level from this sensor.

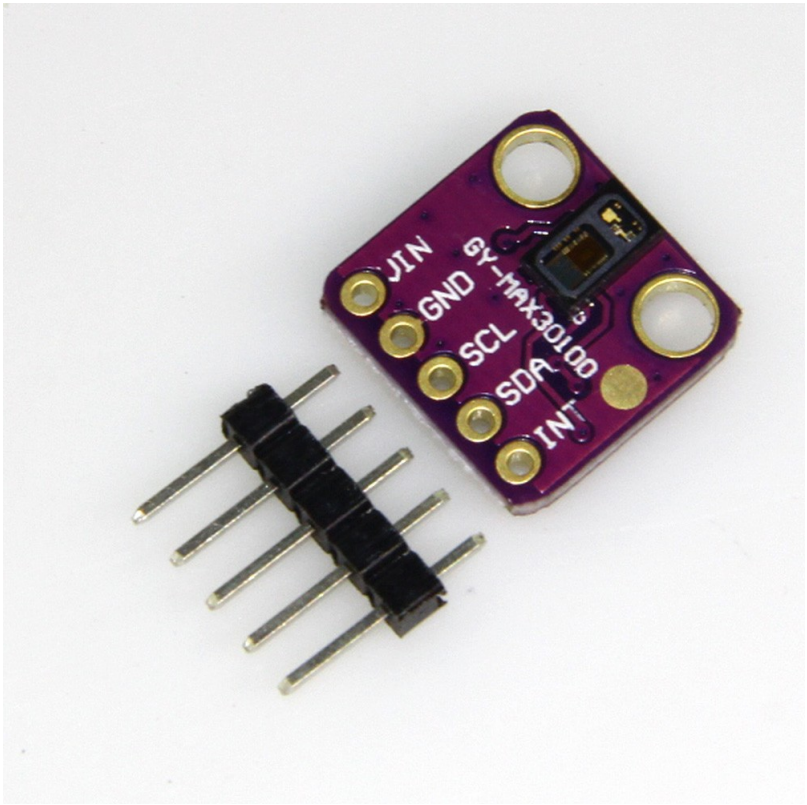


Figure 3.6: MAX30100 Blood Oximeter and Heart-Rate Sensor

Algorithm 2 Sensing the Blood Oximeter and sending the value

```
1: procedure BLOOD_OXIMETER(sensor)                ▷ blood oxygen value
2:     Thread Initialization
3:     I2C communication Initialization
4:     while sensor ==True do                      ▷ Keeping the system running
5:         Get the oximeter value from the I2C module
6:         Send oxygen level value to the firebase
```

3.1.1.3 System for detecting the cough

Cough is one of the most most common symptoms of COVID-19, approximately about 60% of the covid patient have dry cough [30]. The authors (Orlandic, L., Teijeiro, T., Atienza, D., 2020) in the paper [31] collected coughing recording data from the public source and created a machine learning model to detect whether a recording is a cough or not. We have decided to use this trained machine learning model to detect whether the person living in the assisted living facility is coughing or not. The wearable device will continuously record the incoming surrounded sound and send it to the cloud. After that, the system will download the sound from the cloud and pass it to the machine learning model to decide whether the person coughed. The recording will be done in 10-second chunks and after that, the recording will be sent to the cloud. This recording will be done in another thread of the CPU to make sure it doesn't miss any part. For our project USB microphone is used but for the implementation of it in a smart wearable device microphone with a small footprint should be used. In the recording method, we have written the program in such a way that the system will first record 50 seconds of the audio, with a chunk size of 10 seconds, and keep it in the memory. At the fifty-second mark, the recorded audio will be started to send to the cloud. This method is used in order to make a buffer and help the system to work at low-speed internet. A boolean variable will control the system in such a way that the same recording is not sent twice to the cloud.

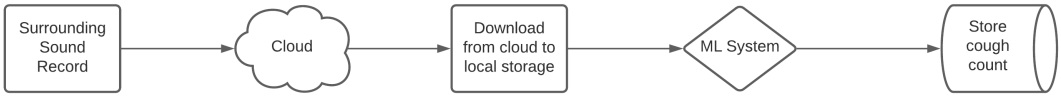


Figure 3.7: Cough Detection Workflow

Algorithm 3 Method For Recording Audio

```
1: procedure RECORD_AUDIO(microphone) ▷ recording audio in .WAV format
2:     Thread Initialization
3:     while record_audio ==True do ▷ Keeping the system running
4:         recording1 = record_audio_for_10_second()
5:         recording2 = record_audio_for_10_second()
6:         recording3 = record_audio_for_10_second()
7:         recording4 = record_audio_for_10_second()
8:         recording5 = record_audio_for_10_second()
9:         send_audio = true ▷ this command starts another procedure in different thread
10:        recording6 = record_audio_for_10_second() ▷ recording is done in six chuns in order to give the system time to upload the audio to the cloud
```

The recording is done in six chunks and is sent one by one. After the ending of the procedure the system again start from the first line of the algorithm and record six chunks. This method is continued while the system runs.

Algorithm 4 Method For Sending Audio

```
1: procedure SEND_AUDIO(memory) ▷ Sending audio in .WAV format
2:     Thread Initialization
3:     while send_audio ==True do ▷ Keeping the system running
4:         send recording 1 to cloud
5:         send recording 2 to cloud
6:         send recording 2 to cloud
7:         send recording 4 to cloud
8:         send recording 5 to cloud
9:         send recording 6 to cloud ▷ one by one all the six chunks of audios are sent to the cloud
10:        Send_audio = false ▷ sending audio is stopped until the next batch of audios recorded
```

3.1.1.4 System for self-reporting of sore-throat and body pain

About 13% of the COVID patients, according to WHO have reported that they have experienced sore throat in the early stage when they were infected with COVID-19 and it is common among elderly people [32]. As a result, sore throat is a vital symptom for early detection of the virus. In addition to that elderly patients who already have an infection in their body are more likely to encounter muscle pain, chest pain, abdominal pain, are an open target for the virus, and are said to be in a very life-threatening stage [33]. Therefore detecting pain is very significant as most elderly people are likely to have several infections in their bodies. Since we will use the non-invasive sensor, it is not possible to detect body pain and sore throat. As these are the most important symptoms to diagnose the COVID-19 disease, the system will have two separate push button switches to make the system aware that he/she is having body pain or sore throat.

Algorithm 5 Method for self-reporting of sore throat

```
1: procedure SORE_THROAT(switch)                ▷ getting value form switch
2:     Thread Initialization
3:     while System ==True do                   ▷ Keeping the system running
4:         if sore_throat_button = True then
5:             send the signal to the cloud
```

Algorithm 6 Method for self-reporting of pain

```
1: procedure SORE_THROAT(switch)                ▷ getting value form switch
2:     Thread Initialization
3:     while System ==True do                   ▷ Keeping the system running
4:         if pain_button = True then
5:             send the signal to the cloud
```

3.1.1.5 System integration

If we program the system without multi-threading then all the sensors have to wait for a particular amount of time to read their data and use the system to send the data, as the code is being executed line by line. On top of that, all the sensor data, for example, audio recording should be done and sent all the time in order to count the number of coughs in a day, therefore if a sensor waits for another sensor in order to collect data, there is a very high chance of data loss from the system. So we cannot allow any sensor to remain idle, with the aim to solve this problem, we have decided to run all the sensors in different threads. This decision will allow the system to independently collect the data all the time. For example, the system will be able to store information like body pain parallel to measuring the body temperature. While doing these two, the system will simultaneously send surround recordings to detect coughing.

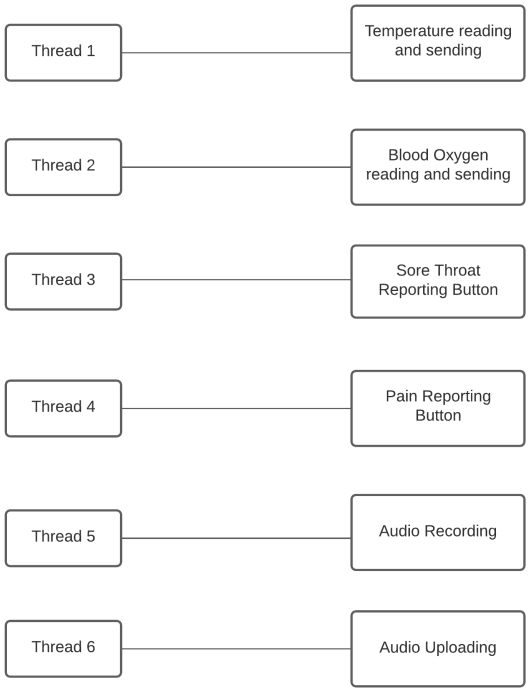


Figure 3.8: Threading Map of Different Sensors

This is the prototype we have developed following all the design parameters that we have discussed above. Our framework proposes a smart wearable device which will be easily used without interfering with the life of older people too much.

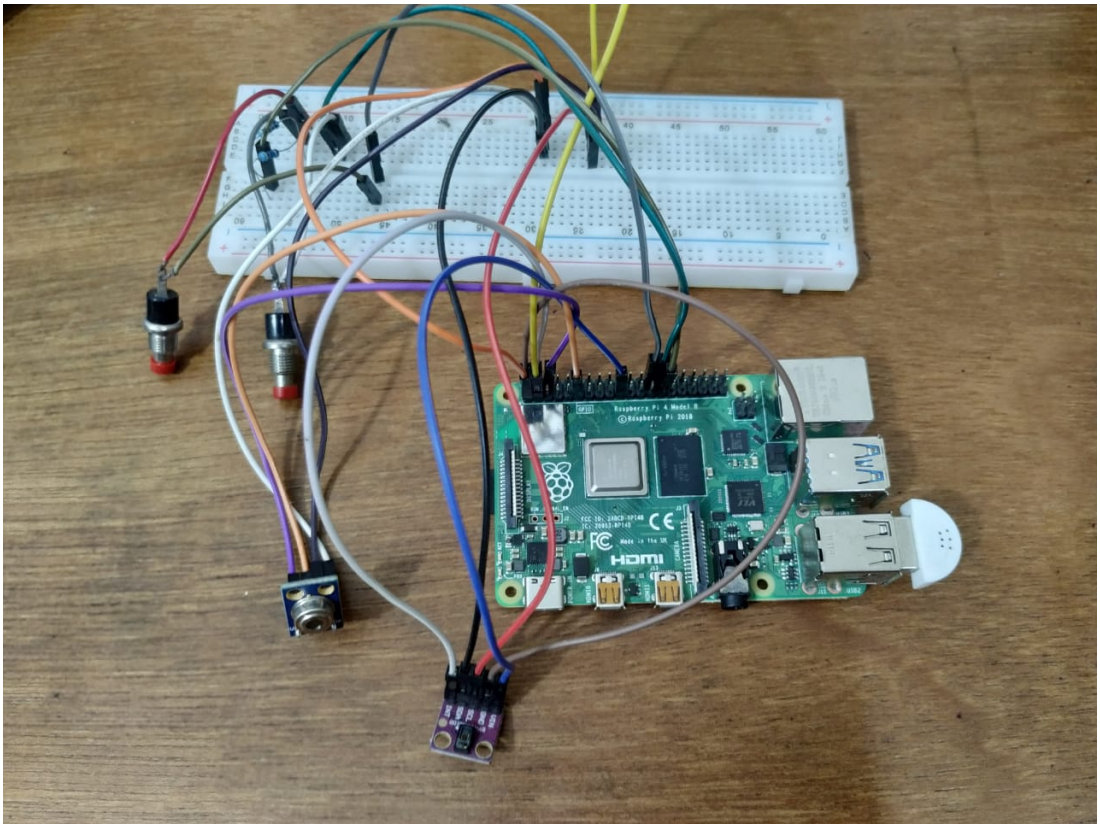


Figure 3.9: Prototype of Wearable System

3.1.2 Framework for collecting the data of remote sensors

Detecting symptoms like runny nose, diarrhoea, and tiredness requires invasive sensors. In order to make the system user-friendly for older people, we have decided to use some remote sensors to detect these symptoms. This part of the system will have the following functions

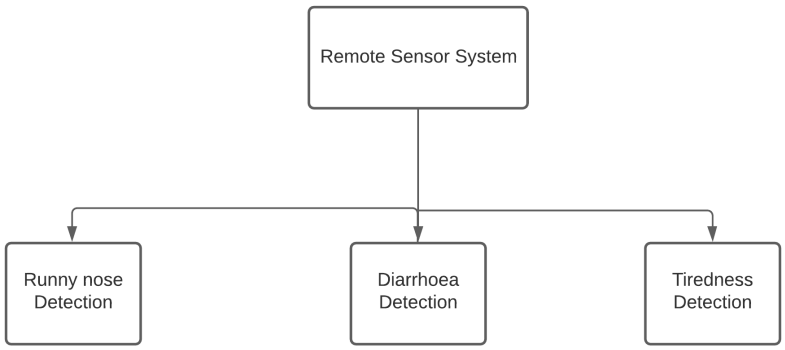


Figure 3.10: Symptoms Detection Using Remote Sensors

For sending the sensor data to the cloud, all the sensors are connected to the ESP32. ESP32 is capable of sending the data to the cloud through its built-in WiFi module, which is the prime reason to select this micro-controller.

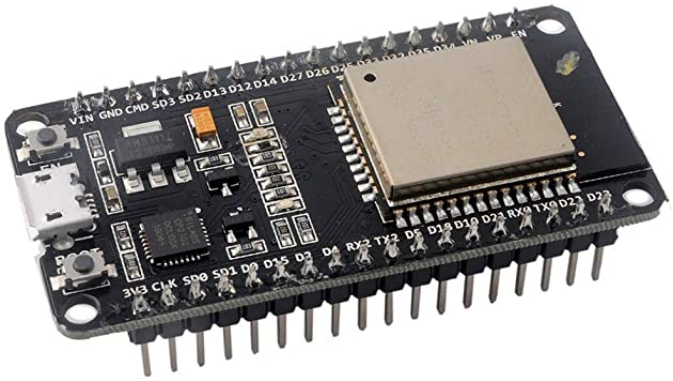


Figure 3.11: ESP32

3.1.2.1 System for detecting the runny nose

A runny nose is one of the most popular symptoms of COVID-19, a recent study [34], has shown that people who have been tested positive are more likely to have this symptom than those who are negative, and it has been reported by the study that it was the second most popular symptom reported by the winter wave. Although this symptom does not occur in the early stage but in COVID-19 hotspots where there are higher COVID-19 positive rates, if an individual has this symptom, then there is a higher chance that it was due to COVID-19 [34]. That is why we have incorporated this symptom in our research. In order to detect whether an elderly individual is having a runny nose, we have designed a smart tissue box system. We have decided to count the number of tissues a person uses to decide whether the person is suffering from a runny nose. All the tissue boxes in the room will have an IR-based counter which will increase the count when a person takes a tissue from the tissue box. The IR counter will be connected to the ESP32, which will send the data to the cloud. After that, the local computer will collect the value from the cloud to diagnose runny nose symptoms.

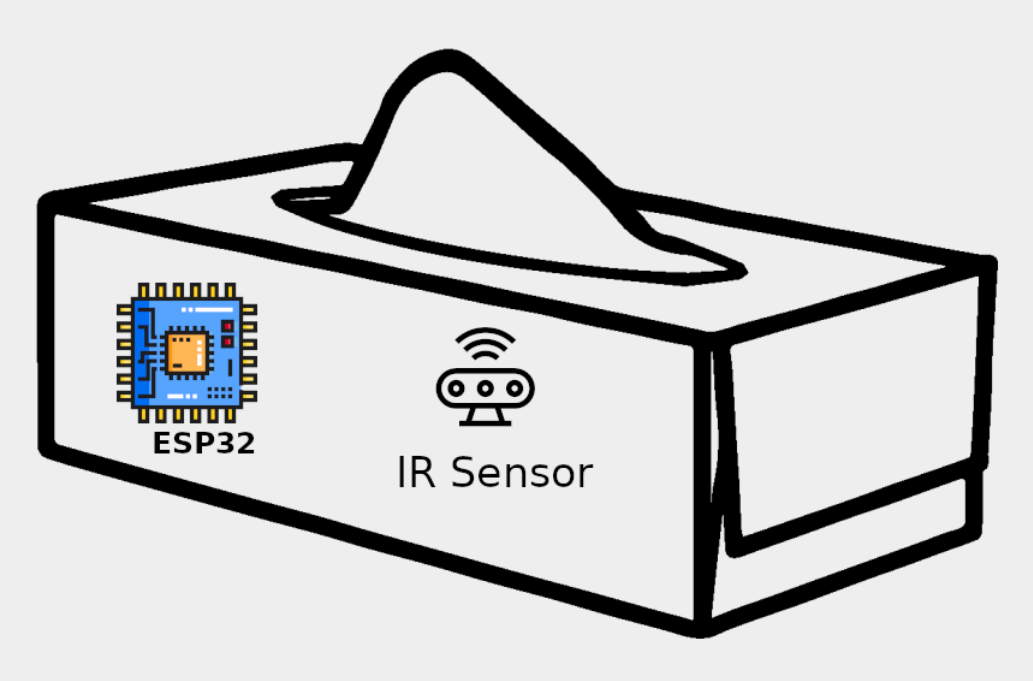


Figure 3.12: Tissue Box Equipped With IR Sensor and ESP32

3.1.2.2 System for detecting the diarrhoea

A study [35], has shown that people who have been COVID-19 positive have faced gastrointestinal symptoms, especially diarrhoea. The study shows that people who have diarrhoea and mild or no respiratory symptoms have been tested positive. So people are unaware, as they prioritize, and are more likely to get tested when they face respiratory illness rather than digestive illness and when they realize it's already too late [36]. According to the study, diarrhea is now one of the major early signs of a person being infected with the virus [35]. So along with the respiratory symptoms, we have mentioned above, we have added this symptom to our research so that every life can be saved. In order to detect diarrhea in a non-invasive way, we will count the number of times a person goes to the toilet. The system counts each entry and exit from the toilet. The system is smart enough to discard all the toilet visits for other causes by comparing the time of entry and exit. If the system detects a toilet visit then it will send the count number to the cloud. Then the local computer will use the value to detect diarrhea symptoms. For detecting the entry and exit, a PIR sensor will be placed closer to the door. When the PIR sensor will detect the movement it will keep a record of entry time and in the same way exit time.

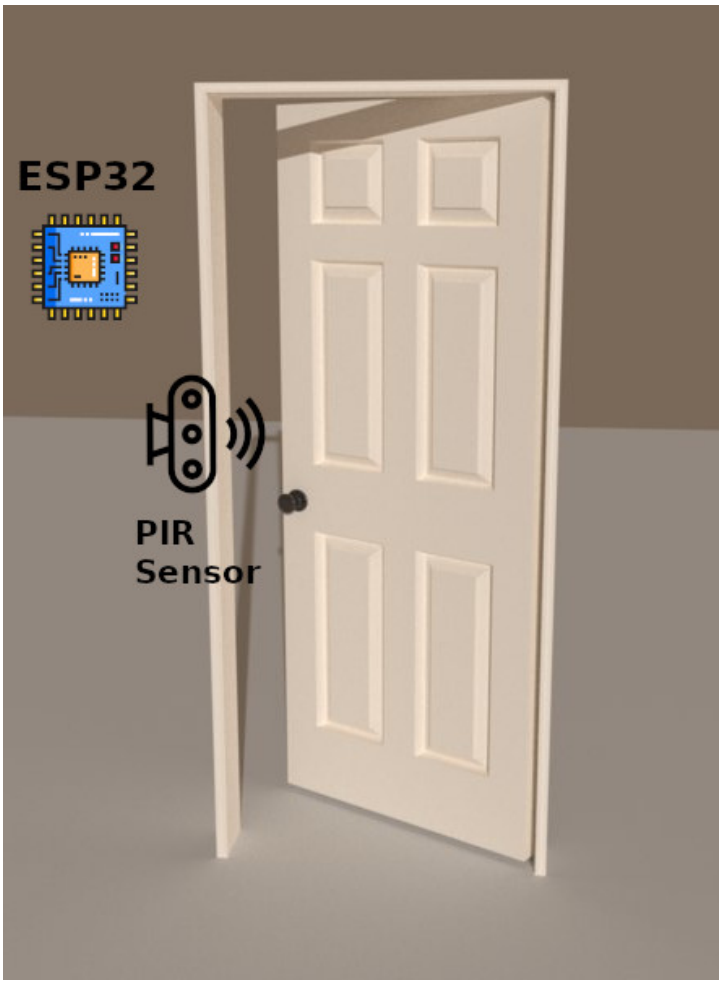


Figure 3.13: Door Equipped with PIR Sensor and ESP32

Algorithm 7 Method for detecting diarrhoea

1: **procedure** DIARRHOEA_DETECTION($entry\_time, exit\_time$) ▷ getting value form PIR sensor
2: Thread Initialization
3: **while** $System$ ==True **do** ▷ Keeping the system running
4: **if** $exit\_time - entry\_time \geq regular\_toilet\_time$ **then**
5: send the signal to the cloud

3.1.2.3 System for detecting the tiredness

Elderly people are generally physically weak, for many people tiredness is one of the very first symptoms [37]. Keeping in mind the physical condition of the old people we have included this symptom. For the purpose of detecting tiredness, we will calculate the number of hours a person is sleeping or staying on the bed. As the surface area of the bed is big, we have decided to use three PIR sensors. If two out of three PIR sensors are occupied, the system will count the number of hours the bed is occupied. The system will regularly send the occupance time to the cloud. From the number of hours the bed is occupied, the tiredness symptom will be detected. The PIR sensor will be placed on the north and south side of the bed as well as either left or right side of the bed.

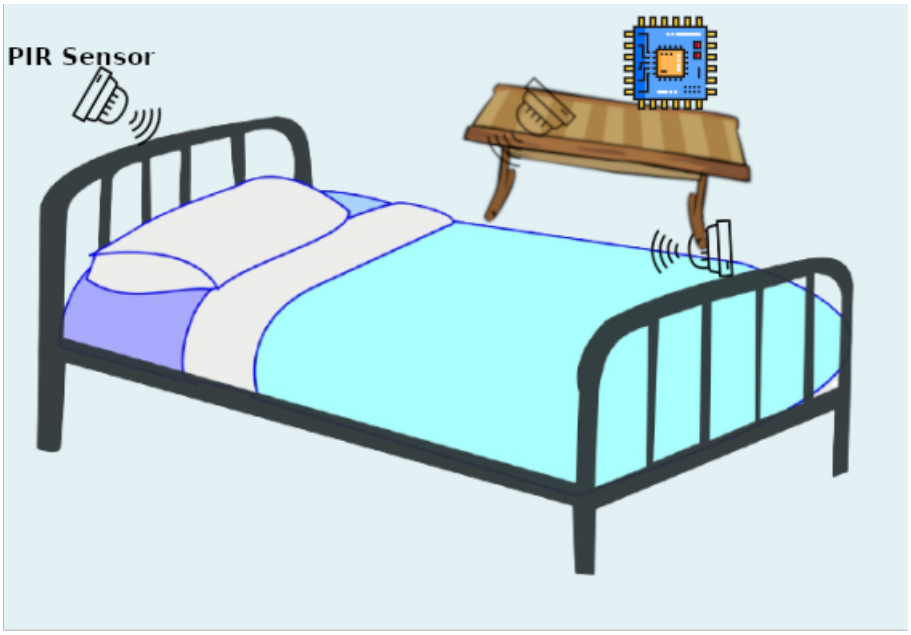


Figure 3.14: BED Equipped with Three PIR Sensor and ESP32

Algorithm 8 Method for detecting tiredness

```
1: procedure TIREDNESS_DETECTION(PIR1, PIR2, PIR3)  ▷ getting value form PIR sensor
2:     Thread Initialization
3:     idle_time = 0
4:     while System ==True do                        ▷ Keeping the system running
5:         if ((PIR1 == 1 and PIR2 == 1) or (PIR2 == 1 and PIR3 == 1) or (PIR1 == 1 and PIR3 == 1) then
6:             idle_time += count_time()
7:             send value to the cloud
```

3.2 Prediction of the COVID-19 from the IoT Sensor Data

The framework can predict COVID-19 using simple symptom attributes like fever, cough, breathing problem, diarrhoea, weakness, and more. As per our literature review, COVID-19 can be predicted in a patient by reviewing the chest X-ray through various machine learning models. However, getting a chest X-ray can be very challenging in this pandemic situation and can take several days. Our proposed model gives older people easy access to the predictability of COVID-19 by analyzing real-time symptoms through IoT devices.

3.2.1 Data Collection

A synthesized dataset was used for the training and validation of the model. Hari H. created the dataset in Kaggle based on WHO guidelines [38]. However, with flexibility, the models can be used on real datasets too. The dataset contains 21 attributes, for example, some COVID like symptoms, continuing diseases (e.g., hypertension, diabetes, asthma, heart disease), previous diseases like chronic lung disease, and other information (e.g., did he/she wore a mask while going outside, did he/she go to crowded areas, did he/she went traveling); and many more. With all attributes, there was also the "COVID-19" feature, which had value in 'yes' or 'no' for a given patient. We selected this feature as our target attribute.

3.2.2 Data Preprocessing and Feature Selection

The features needed for the framework to predict COVID-19 were some of the primary symptoms. The dataset contained some parts that were not relevant to our framework. Also, some of the features gave us some information on the patient's past health history. However, they were not promising to find out the pattern for prediction. So, we got rid of those types of features and continued with features named "Breathing Problem," "Fever," "Dry Cough," "Sore Throat," "Runny Nose," "Headache," "Fatigue," "Gastrointestinal" and "COVID-19". These features had only two unique values for each patient, "yes" or "no." So, we transformed the values to 1 if "yes" and 0 if "no." This way, the whole dataset was binary-encoded for classification and deep learning in an efficient manner.

3.2.3 Splitting Dataset for Training and Validation

To train the models, we need to set the dependent/target feature. We have selected the "COVID-19" feature to be the target attribute. So, we copied all the features except "COVID-19" in the variable "x" and copied the target feature to the "y" variable. Then, to train various machine learning models, we split the dataset into training and test subset. We have used K-Fold cross validation for our model to split the dataset. For the value of "K", we set it to 10; so that, our dataset is divided into 10 splits and for each fitting, all of the 10 folds will be our test subset of the data one after another. Using these shuffles of data, our model will be trained 10 times and the best one will be kept for prediction. Data standardization is needed for machine learning models as a means of data preprocessing for numerical data. So, we imported another function from scikit-learn's library, "StandarScaler." We created an instance for the function and transformed the newly created variables after splitting the dataset, "x_train" and "x_train." Now, the dataset is ready for training and validation. For validation, "Accuracy" is the most popular metric. However, a model's performance cannot be judged based only by the accuracy. So, we have used other metrics, such as - precision, recall, and f-score with all the 10 folds of our data. The metrics were calculated using the mean of metrics for all the folds.

3.2.4 Model Selection

The dataset contains binary-encoded data for each attribute and each tuple of patients. We know that classification models perform better for binary data than other machine learning models. So, we fitted some of the most popular classification models like – Decision Tree Classifier, Random Forest Classifier, etc. We also tried one of the regression models, logistic regression. Additionally, we fitted a performance-oriented decision tree model, XGBoost. Later on, we trained SVM classification model. Finally, we proceeded to basic deep learning to get more robust results from machine learning.

3.2.4.1 Logistic Regression Model

The author says that regression analysis is used for predicting the connection between independent tuples and the target [39]. Linear regression in statistics is what made the regression model possible. The linear regression gives us the curve for the relationship, which is very useful for predicting various issues. The supervised classification algorithm logistic regression is a regression model that predicts outcomes on a binary basis, says Thanda [39]. The logistic regression model works very well on categorical data. However, it also works on continuous and nominal data. The linear regression uses MSE (Mean Squared Error) as its cost function [39], and for logistic regression, the cost function graph generates a non-convex function of "theta"; where theta is a collection of parameters, say Swaminathan [40].

The fundamental function of logistic regression is logistic function. This can take the input from datasets and convert it into values ranging from 0 to 1.

$$1/\left(1+\mathrm{e}\text{^}-value\right) \tag{3.1}$$

Logistic regression can be used to classify using both the continuous data and discrete data. The output depends on the input and co-efficient like an example below where x is input and y is output

$$y = e\text{^}\left(b0+b1*x\right)/\left(1+e\text{^}\left(b0+b1*x\right)\right) \tag{3.2}$$

Although this is a classifier, in beneath this is a predictor. It predicts the chances of being on of the classes. One of the advantages of the logistic regression is it can classify by considering many types of input variables. For example, output Y can depend on inputs X1, X2, X3, etc. [41].

$$\mathrm{P(X)} = \mathrm{P(Y \mid X1, X2, X3)} \tag{3.3}$$

The cost function of linear regression is

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(y_i - (mx_i + b))^2$$

- N is the total number of observations (data points)
- $\frac{1}{N}\sum_{i=1}^{n}$ is the mean
- y_i is the actual value of an observation and $mx_i + b$ is our prediction

3.2.4.2 Decision Tree Classification Model

The author defined the decision tree model as a supervised learning algorithm where we start from the root node and branch out using comparison to predict the target variable [42]. The decision tree model is very beneficial where the problem is either classification type or regression type. Decision tree classification is used when the data is categorical (e.g., "yes," "no"; "1", "0", "True," "False") while decision tree regression is used if the data is continuous. A decision tree has a root node from which the data is split into more nodes using branches. The branch is where the decision is made upon the value of the record. For the whole dataset, this procedure is continued, leading us to the ending nodes called leaf/terminal nodes. If we pick any node other than the root node or leaf node, we will see a sub-tree of some depth. There is a variety of decision tree algorithms to meet the requirements of a dataset. Some are - ID3, C4.5, CART, CHAID, MARS, says Chauhan N. [42]. Generally, a decision tree model is unparameterized, but there is a problem deciding which node to be the root node. If we randomly choose one, then the accuracy may get hampered. So, "criteria" may be used as a parameter. There are some options to choose from; they are - Entropy (uses the measurement of randomness), Information Gain (branches utilizing the measure of wellness), Gini Index (uses Gini as the cost function to evaluate the branching), Gain Ratio (chooses features with more significant value to be considered as the root node), Reduction in Variance (chooses the best way to split data through selecting branch with a lower variance; only used in decision tree regression problems), Chi-Square(finds the importance between the differences between the immediate root node and its children nodes) [42]. Below is given the generalized formula for each criterion.

Formula for "Entropy" of one feature [42]

$$E(S) = \sum_{i=1}^{c} -p_i \log_2 p_i \tag{3.4}$$

Formula for "Entropy" of multiple features [42]

$$E(T, X) = \sum_{c \in X} P(c) E(c) \tag{3.5}$$

Formula of "Information Gain" calculation using entropy [42]

$$\textbf{Information Gain(T,X) = Entropy(T) - Entropy(T, X)} \tag{3.6}$$

Formula for "Gini index" Criterion [42]

$$\text{Gini} = 1 - \sum_{i=1}^{c} (p_i)^2 \tag{3.7}$$

Formula for Measurement of "Gain Ratio" Criterion [42]

$$GainRatio = \frac{InformationGain}{SplitInfo} = \frac{Entropy(before) - \sum_{j=1}^{K} Entropy(j, after)}{\sum_{j=1}^{K} w_j \log_2 w_j} \tag{3.8}$$

Formula for "Reduction in Variance" Criterion [42]

$$Variance = \frac{\Sigma(X - \overline{X})^2}{n} \tag{3.9}$$

Mathematical Representation of "Chi-squared" Criterion [42]

$$\chi^2 = \sum \frac{(O - E)^2}{E}$$

Where:
χ^2 =Chi Square obtained
O =observed score
E =expected score

However, in the worst case, we may still overfit the model, which can be solved by pruning the dataset or using a modified version of the decision tree model, the Random Forest algorithm.

3.2.4.3 Random Forest Classification Model

To define the random forest model, Donges N. said that random forest is one of the supervised machine learning models where instead of a single decision tree, an ensemble of decision trees is built to predict the target [43]. Because of the creation of an ensemble of decision trees, it is called the random forest model. The random forest creates decision trees using the "bagging" method, which uses various learning algorithms and merges them for better predictions. Like the other two learning models described above, random forest is used in both classification and regression problems. This model searches for a better algorithm that is the best fit for the random subset of the attributes to get a better result. There are some benefits to use the random forest model. The overfitting problem that we face in the decision tree is solved in the random forest model. However, overfitting may still arise in the random forest model with few decision trees. The scikit-learn's library for the random forest also offers a table to display the importance of all attributes. So, we can find which attributes contributed more to the prediction of the model.

Inorder to determine how the nodes would branch, on a classification data, the Random Forest uses the Gini index:

$$\text{Gini} = 1 - \sum_{i=1}^{C} (p_i)^2 \tag{3.10}$$

In this Equation, pi stands for relative frequency of a class and c stands for number of classes. By using this equation the algorithm decides which branch to choose next with the help of the class and probability to calculate the Gini of each branch of a node [44].

Besides the Gini index, the Random Forest algorithm also uses the Entropy to figure out the branching of the nodes in a decision tree:

$$\text{Entropy} = \sum_{i=1}^{C} -p_i * \log_2 (p_i) \tag{3.11}$$

The probabilities of certain outputs are used to figure out how the branching of the nodes would occur [44].

3.2.4.4 Support Vector Classicfication Model

McGregor M. illustrates that SVM(Support Vector Machine) is a supervised learning method that learns to separate the data points by chooses the decision boundary to get the highest distance from the closest data points of all the classes [45]. Support vector machine works with both classification problems and also with regression problems. The classifier of the SVM is called SVC(Support Vector Classifier). SVM chooses the best separator to classify the vectors or data points; so, the SVM algorithm works better than some of the better machine learning algorithms like

K-nearest neighbors [45]. The basic working principle is to differentiate the data points based on the decision boundary. The differentiation can be a simple line to a collection of hyperplanes. Generally, there are two types of SVMs. The simple SVM uses lines for linear classification and regression. And, the kernel SVM separates the data points by fitting one or more hyperplanes. There are some of the kernel functions available to use with the model. The most popular and practiced SVM kernel is the linear kernel. When it comes to text classification, it generates a better result with efficiency [46]. The kernel is best suited for situations where there are many features to get to a prediction. The kernel formula for linear kernel is,

$$F(x, xj) = sum(x \cdot xj) \tag{3.12}$$

x and xj are data to be classified [46]
Gaussian Radial Basis Function or RBF is the most preferred kernel when using SVM models, and the data is non-linear [46]. In the kernel function, the gamma ranges between 0 and 1, most preferably, 0.1. The kernel function for RBF is given below.

$$F(x, xj) = exp\left(-gamma * \|x - xj\|\hat{}2\right) \tag{3.13}$$

x and xj are data to be classified [46]

SVM uses the sigmoid function as a kernel function [46]. The function resembles the neural network of two layers of the perceptron model. The kernel function for the sigmoid kernel is given below,

$$F(x, xj) = tanh(\alpha x a y + c) \tag{3.14}$$

α is an weight function and x, xj are data to be classified [46]

The polynomial kernel demonstrates a more generalized linear kernel. In comparison to other kernels, the polynomial kernel is less accurate and efficient. The kernel function is given below,

$$F(x, x]) = (x \cdot xj + 1)\hat{}d \tag{3.15}$$

"." denotes dot production
"d" denotes the degree
x and xj are data to be classified [46]

There are other types of kernel available to meet the complexity of a machine learning problem. Some are-

Gaussian Kernel Formula [46]:

$$k(x, y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) \tag{3.16}$$

Bessel Kernel Formula [46]:

$$k(x, y) = \frac{J_{v+1}(\sigma\|x-y\|)}{\|x-y\|^{-n(v+1)}} \tag{3.17}$$

Anova Kernel Formula [46]:

$$k(x, y) = \sum_{k=1}^{n} \exp\left(-\sigma\left(x^k - y^k\right)^2\right)^d \tag{3.18}$$

The kernel functions generally fix the problem where the number of attributes is larger than the total data points. In that case, the model can be over-fitted and, hence, the kernel functions and regularization are necessary [45].

3.2.4.5 XGBoost Classification Model

XGBoost is a "state-of-the-art" machine learning model that uses gradient boosting to provide better results from other machine learning algorithms, says Pathak M. [47]. According to Morde V., this model is a decision-tree-based ensemble machine learning model [48]. The boosting that he talked about works on ensemble principle by combining a group of weak learning algorithms and output accuracy. Outputs are weighted, and a small weight is given to get more correct predictions. The central concept behind the XGBoost model is that outputting a summary of feature importance and parameters and using the summary on a weak model gives us a better and more robust model. XGBoost has become very popular since it was introduced. Pathak M. also mentioned the reasons behind the model's popularity [47]. The model is written in the C++ programming language, which makes the XGBoost algorithm better than other ensemble classifiers. XGBoost also can utilize the power of multi-core computing as the core algorithm of it is parallelizable. The model offers various adjustments through parameters (e.g., cross-validation regularization, missing values, tree parameters, etc.). While creating an instance, a user can set hyperparameters. Among them, "learning rate" ranges between 0 to 1 and denotes how fast or slow the model learns. The "max_depth" determines the depth

of a tree. The "n_estimators" indicates the number of trees to be used. Through "objective" hyperparameter, the model allows us to set for what objective we want to use the model. Three types of objectives are available, "reg: linear" for linear regression, "reg: logistic" for logistic regression and, "binary: logistic" for binary classification problems with probability [47].

Inorder to build trees, the XGBoost algorithm applies the Loss Function, by reducing the value below:

$$\begin{gathered}\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \\ \text{where } \Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2\end{gathered} \tag{3.19}$$

The first part of the equation determines the Loss Function and the second part determines the regularization term lambda. w stands for the leaf weights, T stands for the number of terminal nodes and gamma stands for the user-definable penalty [49].

As described earlier in order to find the output XGBoost algorithm, it uses the previous prediction (i-1)th and the output of the i-th tree. By using this idea the first part of the above equation 3.19 is modified as below [49]:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)\right) + \Omega(f_t) \tag{3.20}$$

With the aim to get an optimal output value for this equation, for Classification XGBoost uses a second order Taylor Approximation and the above equation is optimised to [49]:

$$\tilde{\mathcal{L}}^{(t)}(q) = -\frac{1}{2}\sum_{j=1}^{T} \frac{\left(\sum_{i \in I_j} g_i\right)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma T \tag{3.21}$$

gi stands for the first derivative of the loss function and the hi stands for the second derivative of the loss function

In order to split the branches of the trees XG Boost chooses the branch that would result in maximum Loss Reduction [50], so the function for calculating this is given below [49]:

$$\mathcal{L}_{\text{split}} = \frac{1}{2}\left[\frac{\left(\sum_{i \in I_L} g_i\right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left(\sum_{i \in I_R} g_i\right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left(\sum_{i \in I} g_i\right)^2}{\sum_{i \in I} h_i + \lambda}\right] - \gamma \tag{3.22}$$

3.2.4.6 Deep Neural Network

According to Moolayil J., a deep neural network or deep learning model is a machine learning approach to solve or predict problems with algorithms inspired by human brain cells [51]. The concept to use the principle of the mechanism of a human brain cell network raised when the problem was very easy for the human mind—for example, classifying an image, whether it is a cat or dog, an audio clip whether the voice is of male or female. Popular machine learning algorithms were not very accurate when it comes to predicting these types of problems.

The deep neural network consists of layers of neurons, as can be seen in a human brain. Like the human brain, the neurons can also have connections among themselves. These neurons can pass information based on the given input. Whether the data will be passed or not is decided on calculating a function called "activation function." The calculation of the "activation function" is that the output of the current neuron will be passed to the next neuron if the incoming neurons give a higher value than the threshold. Also, with each neuron, an weight is assigned to balance the whole network. So, a simple neural network will have inputs, weights, input function, activation function, and output. This model of a simple neuron is called a perceptron.

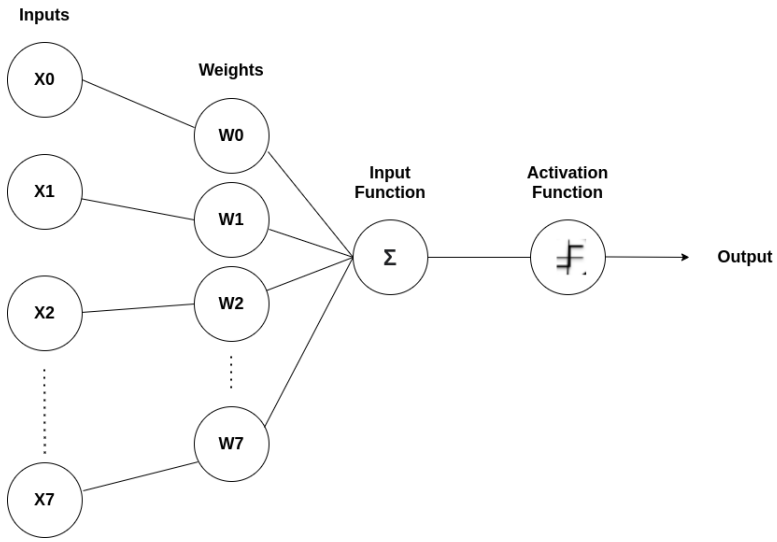


Figure 3.15: A Perceptron Model

A deep neural network is built with layers of perceptrons. The deep neural network has one or more hidden layers between input and output layers. The hidden layers are connected and used to learn prediction. Deep learning provides feedback using an algorithm called "backpropagation."

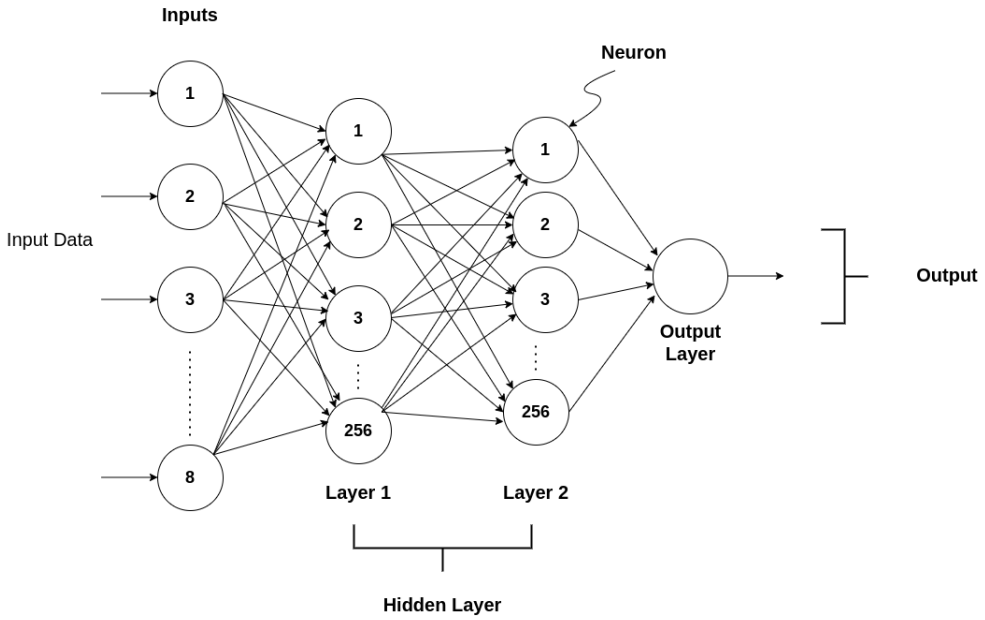


Figure 3.16: Deep Neural Network

Chapter 4

Implementation and Result Analysis

This section has two main parts. In the first section, we will discuss about the implementation of the Machine Learning and Deep Learning models discussed in the Methodology section then in the second section we will analyze those results.

4.1 Implementation of the Machine Learning and Deep Learning models

In this section we will discuss the implementation of the Logistic Regression, Decision Tree Classifier, Random Forest Classifier, Support Vector Machine Classifier, XGBoost Classifier, and Deep Neural Network.

4.1.1 Implementation of Logistic Regression

To implement a logistic regression model, first, we imported the "LogisticRegression" function from the scikit-learn's library and created a default instance of it. Then we fitted the model with the "x_train" and "x_test" that we created earlier. We tested the model with the "x_test" values and got predicted outcomes. Using the "scikit-learn.metrics" library, we then calculated the confusion matric and accuracy. The accuracy that we got from this model was 91.17%.

4.1.2 Implementation of Decision Tree Classification Model

The implementation for the decision tree classifier is almost similar to what we did earlier. The only exception is that we had to import the "DecisionTreeClassifier" function from scikit-learn's library rather than "LogisticRegression." Also, we calculated the accuracy and confusion matrix as we calculated for the logistic regression model. With the decision tree classifier, we got around 92.94% accuracy.

4.1.3 Implementation of the Random Forest Classification Model

The implementation of the random forest classification is almost similar to the logistic regression and the decision tree classification models. The random forest classifier offers more versatility and flexibility when it comes to fitting a model. For instance, when initializing an instance of "RandomForestClassifier" from the scikit-learn's library, we can set various hyperparameters that can lead us to a better result. For our model, we have set the "n_estimators" hyperparameters to "40". The "n_estimators" hyperparameter increases the predictability power of the model because of the creation of a greater number of decision trees. After we have fitted the model using the preprocessed data, we calculated the accuracy and confusion matrix as we did for the other models. With random forest classification, we got nearly 93.30% accuracy.

Later, we have printed the random forest feature important. In the table and graph, we can see that the four major features of the dataset that have contributed more to the prediction were, respectively, "Breathing Problem," "Dry Cough," "Sore Throat," and "Fever."

Feature	Importance
Breathing Problem	0.261
Dry Cough	0.258
Sore Throat	0.192
Fever	0.148
Fatigue	0.042
Gastrointestinal	0.039
Running Nose	0.036
Headache	0.025

Table 4.1: Tabular Representation of Feature Importance

4.1.4 Implementation of Support Vector Classification Model

We imported the "SVC" function from scikit-learn's library and created an instance of it. When creating the instance, we set the kernel as "linear" as it gives better results when the problem requires classification and gave the instance a pseudo-random value of "0" for the current shuffling. Then we predicted the outcomes for the "x_test" validation subset of data and calculated confusion matrix and accuracy likewise. With SVM classifier, we got an accuracy of 92.05%.

4.1.5 Implementation of XGBoost Classification Model

We imported the "xgb" function from the scikit-learn's library and initiated an instance of the function. As hyperparameters, we have set "objective" as "binary: logistic" as our preprocessed data consists of binary data, and the problem falls into a classification problem. The XGBoost model shows us the validation error while the model learns something, and, in the Output tab, we see that till approximately

235 trees, the validation error was the lowest. After a while, the validation error started to increase again, informing us that the model has begun to overfit. So, we have set the "n_estimators" hyperparameters to 235. When fitting, we have set the evaluation metric to be "error" to get the validation error after processing each decision tree. The "error" metric is specific for binary classification. We have also set the "verbose" to "true" to print the validation error in the output pane to more precise information about what is happening inside. Lastly, we calculated the accuracy and the confusion matrix the same way we did for the other models. After fitting the model, we got an accuracy of 93.01%.

Feature	Importance
Sore Throat	0.455
Dry Cough	0.165
Breathing Problem	0.155
Fever	0.099
Gastrointestinal	0.040
Fatigue	0.036
Running Nose	0.028
Headache	0.022

Table 4.2: Tabular Representation of XGBoost Feature Importance

Like random forest classification, we then printed the feature importance. From the table and graph, we can see that "Sore Throat" is the primary attribute contributing the most to the prediction. However, "Dry Cough," "Breathing Problem," and "Fever" also contributed to some extent.

4.1.6 Implementation of Classification with Deep Learning

The deep neural network can be implemented through various deep learning frameworks. While "TensorFlow," "MxNet," "PyTorch" are some of the low-level frameworks, "Keras," "Gluon" are examples of high-level frameworks [51]. We have chosen the "Keras" framework, which uses "TensorFlow" as a backend. So, we have imported "keras" from the "tensorflow" library, and, from the "keras's" library, we have imported "layers" and "callbacks." Then we declared a sequential model where the number of hidden layers, activation function, etc., can be set. The dataset is relatively smaller in size. So, we have decided to create one input layer, two hidden layers, and one output layer. At the start and in between each layer, we have added batch normalization layers for internal scaling. The input and hidden layers contain 256 neuron units, and the activation function is set to "ReLu" (Rectified Linear Unit). After each of these two layers, we have dropped out 30% of the neurons contributing less to the learning process. The "Keras" framework decides which neurons to drop internally. The output layer consists of only one neuron, and the activation function is set to "Sigmoid." As a result, we can have a classification with probability. Then we compiled the model using some hyperparameters. For

an efficient deep learning model, we need to set the learning rate in a way that it is not very slow or not so high; otherwise, the model can be under-fitted or over-fitted. However, optimizers can reduce the hassle; and so, we have set the optimizer to "adam," which sets the learning rate adaptively. Then we put the loss function to "binary_crossentropy" and the metrics of the loss function to "binary_accuracy." We do not know for how many epochs the model will keep reducing the validation error. If we set the number of epochs larger, the model will surely get over-fitted. That is why we have used the early stopping process, which stops the model when no more improvement is achieved from the model. We have set the threshold, "min_delta" to 0.0001, "patience" to 5, and "restore_best_weights" to "True," which means that the model will try to achieve at least 0.0001 improvements after each epoch, and when the model fails, it will wait for five more epochs to check for any further progress. If no further improvement is not found, the model will restore the best result that it got. Then we fitted the model with "x_train," "y_train" data, and "x_valid," "y_valid" validation data. The model was fed 512 rows each time using the "batch_size." We have set the epoch size larger than needed because the model will stop when overfitting occurs.

The model ran for approximately 65 epochs to reduce the training loss as much as possible. The early stopping saved the model from being over-fitted. Also, we can see that the binary accuracy of the model is 94.83%. We need to validate if the model was successful or not. For this, we have plotted loss vs. epoch graph named "Cross-entropy" and accuracy vs. epoch graph "Accuracy." If everything is working well, the training loss should be lower than the validation loss, and the training accuracy should rise, coping with the validation binary accuracy.

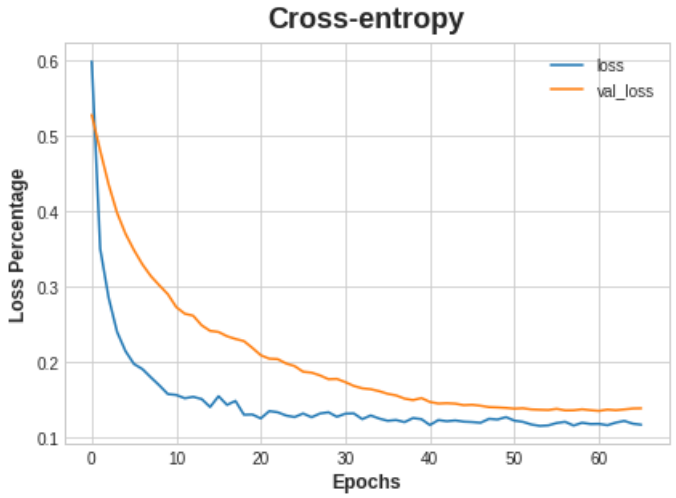


Figure 4.1: Cross-entropy for the Deep Neural Network

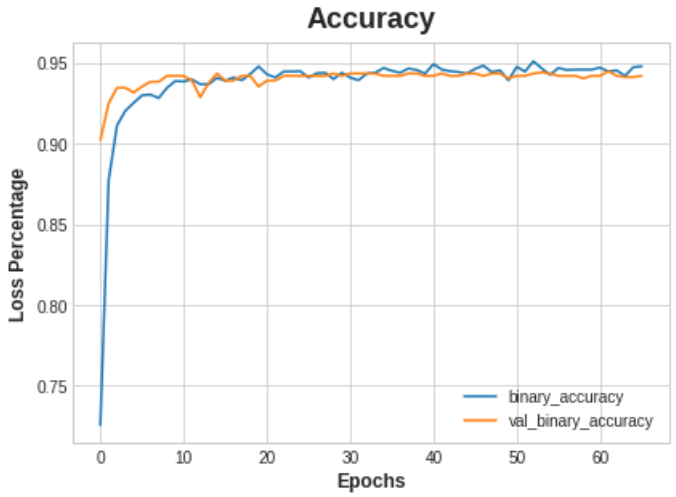


Figure 4.2: Accuracy for the Deep Neural Network

From the two graphs above, we see that the training loss fell faster and stayed under the validation loss. Also, the training accuracy was higher than the validation binary accuracy. So, the model was fitted successfully.

4.2 Analysis of the Machine Learning and Deep Learning models

The performance of a model is measured and compared when it comes to choosing which of the model gave better prediction. We have calculated accuracy for each of the models previously. However, the accuracy of a model gives us a general sense of how a model has performed. To get a better understanding, we need other metrics and a comparison of those metrics among them. The "confusion_matrix" function from scikit-learn's library helps in this regard. According to the author, the function outputs four aspects of the prediction: True Positive, False Positive, False Negative, and True Negative [52].

When the actual data is positive, and the model predicts the data as positive, we call those predictions true positive. Likewise, we can define true negatives for negative predictions. We call the predictions false positive if the actual data was negative but the model wrongly predicted as positive. Similarly, false negative predictions are when the actual data was positive, but the model interpreted them as negative. All of the aspects are very important as they help us to evaluate the models using various measurements.

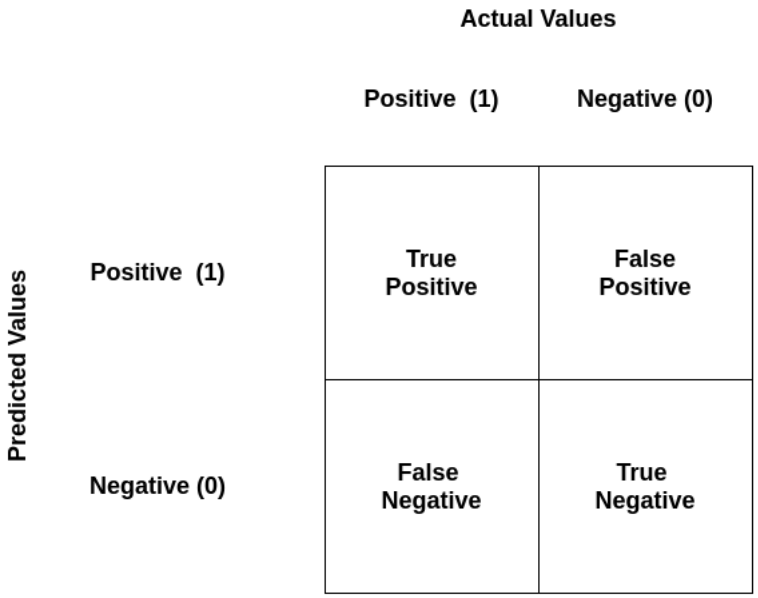


Figure 4.3: Distribution Table for True Positive, False Positive, False Negative, True Negative

With a confusion matrix, we can evaluate four measurement metrics, which are - accuracy, precision, recall, f-score [52]. The accuracy denotes the percentage of total correct predictions out of total predictions. It can also be measured using the confusion matrix.

$$Accuracy = \frac{CorrectPredictions}{TotalPredictions} \tag{4.1}$$

Using confusion matrix, let us say TP=True Positive, TN=True Negative, FP=False Positive, FN=False Negatvie

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.2}$$

Precision indicates how many times positive predictions were actually positive. This can be calculated using the confusion matrix.

$$Precision = \frac{TP}{TP + FP} \tag{4.3}$$

Recall determines, out of all positive data, how many times the model could predict positiveness. There is a formula too for recall.

$$Recall = \frac{TP}{TP + FN} \tag{4.4}$$

For most of the time, models cannot be compared using only precision and recall. So, to measure them together, we can use the harmonic mean of precision and recall [52]. This metrics is called f-score, which can also be calculated using the confusion matrix.

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4.5}$$

We have compared all the models upon these four metrics after using K-Fold Cross Validation. However, instead of calculating them manually, we have used the respective functions from the scikit-learn's metrics library. After calculating all of them, we have created a table that summarizes all the models together.

Model	Accuracy	Precision	Recall	F-Score
Deep Neural Network	0.948285	0.947767	0.990426	0.967982
Random Forest Classification	0.933042	0.949371	0.970868	0.958565
XGBoost Classification	0.930109	0.938164	0.978140	0.957688
Decision Tree Classification	0.929360	0.951504	0.961752	0.956535
Support Vector Classification	0.920539	0.929968	0.975401	0.952065
Logistic Regression	0.911719	0.918184	0.978144	0.947162

Table 4.3: Classification Report for Models

From the table, we can see that the deep neural network has performed better in every measurement aspect. The Deep learning model was followed by Random Forest Classification, XGBoost Classification, and Decision Tree Classification. In terms of accuracy and F1-score, they were very competitive. As the deep learning model was proven to be the best algorithm for our scenario, we decided to predict COVID-19 using this model to get a robust prediction.

4.3 Prototype Evaluation

In this section we will discuss about the data preprocessing of the values coming from the sensors, how and on which basis they are taken as a symptom and finally data validation of the fever and breathing difficulty using Hidden Markov Model.

4.3.1 Preprocessing of Sensor Data

Each record of the 10-minute window contains a real-time reading from the IoT sensors. In that 10-minute window, we get values from the sensors in milliseconds. So, in that window, we will have too much information to converge into any prediction. For this reason, we need to reduce the dimension in a way so that each 10-minute window will represent only one tuple of the total data received. Now, if we consider a window of one hour, we will have six dimensions-reduced tuples. This way, the prediction will be more generalized. To implement, we have used the feature extraction process. We have considered three measurements for the central tendency for feature extraction: mean, median, and mode.

Mean is the mathematical average of the values. If we have "n" number of values, then the mean of the records will be the summation of all the values and division of the result by "n." Mean is best for the continuous data. The median data is divided into half and finds similar records above the smallest value and below the largest value [53]. The median is best suited for skewed data where the data is continuous but not symmetrical (has a tail in one end). Mode is used when that data is categorical. It finds the value that occurred mostly in the data. The sensor values are skewed; so, we will be using the median for feature extraction.

There are 8 sensor values that we are working with. Among them the temperature sensor will tell us whether the person has fever or not. If the temperature is above 38 degree Celsius then a person has fever [54]. Although oximeter is not the most reliable way to measure the breathing difficulty, if the person's oxygen saturation level is below 89%, we will consider the person is having breathing difficulty [55]. If the person presses the sore throat button then the system will understand the person is having sore throat, same thing will be considered for the pain button which will be considered as headache. The number of tissues a person using will be considered to detect whether a person is having any runny nose symptom or not. As there are no official number of tissues a person suffering from runny nose uses, we are considering 15 as a threshold value. The dry cough will be counted based on the audio recording and machine learning model. As there are no official data of the number of times a healthy person coughs, we consider a person is suffering from dry cough if he/she coughs more than 10 times an hour. According this article [56], a person aged 65 or above should sleep 7-8 hours daily. If the occupancy sensor of our framework detects a person is sleeping or staying on the bed more than 10 hours then we will consider the person is weak or fatigued. According to this article [57], a person suffering from diarrhoea will have 3 to 4 times of bowel movement. If the PIR sensor of our framework detects a person going to toilet for more than 4 times a day, the system will consider that the person is suffering from gastrointestinal problem.

However, there may arise some situations where the data can show some inconsistency. For instance, the "Fever" feature can have unwanted 1s and 0s. Let us say the target person has a fever for one hour, but for some reason, any of the windows can give us information that the person does not have any fever. The target may take a bath, and the body temperature may fall below the threshold. However, the target is still suffering from fever. An opposite scenario can also occur. The target may not be suffering from fever, but his/her body temperature may rise above the threshold. For this reason, the sensor data can lose its robustness. The "Difficulty-in-Breathing" can also have this kind of issue. Due to the long time use of the oximeter sensor, it can give us some inconsistent data. The target person may not be having breathing difficulty, but the sensor can provide us with the saturated output. So, the "Fever" and the "Difficulty-in-Breathing" feature have to be validated. For this, we have chosen the "Hidden Markov Model" algorithm.

4.3.2 Hidden Markov Model

The author defined the hidden Markov model (HMM) as a derivative of the Markov chain model [58]. The Markov chain model is used for making a rigid prediction on a sequence by evaluating current states. However, the current states and the observable states may remain hidden. To eradicate this problem, the hidden Markov model was introduced. According to the author, the model contains the following elements.

A set of states to "N,"

$$Q = q_1 q_2 \dots q_N \tag{4.6}$$

Transitional probability matrix that represents the probability of transition from each state to other states,

$$A = a_{11} \dots a_{ij} \dots a_{NN}$$

here, a_{ij} represents the probability of transition from state "i" to "j"

The sequence of observations to "T,"

$$O = O_1 O_2 \dots O_T \tag{4.7}$$

The emission probability representing the probability of an observation from state i,

$$B = b_i(O_t) \tag{4.8}$$

The initial probability of all the states,

$$\pi = \pi, \pi_2, \ldots, \pi_N \tag{4.9}$$

According to the author, the HMM model is used for three types of problems: Likelihood, Decoding, and Learning [58]. In the first problem, the HMM is used to determine the likelihood of an observation sequence. In an HMM model, λ = (A, B), the probability P(O|λ) is calculated; here, O denotes the observation sequence. This problem uses a forward algorithm to propagate, and the forward algorithm contains three steps.

Initializing step,

$$\alpha_1(j) = \pi_j b_j(o_1)\,; 1 \leq j \leq N[T2] \tag{4.10}$$

Recusion step,

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)\,; \quad 1 \leq j \leq N, 1 < t \leq T \tag{4.11}$$

Terminating step,

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{4.12}$$

The decoding problem is where the job is to find a sequence of variables that is internally powering the observation sequence [58]. This way, we can find the best-estimated state sequence, Q = q1q2q3 ...qT through the observation sequence, O = o1,o2,...,oT in an HMM model of λ = (A,B). The decoding is done through the "Viterbi" dynamic algorithm. Like the likelihood problem, it also has three steps. The "Viterbi" algorithm also returns the best path; hence, for each step, we have to calculate two things, the "Viterbi" and the path to find the best probability.

Initializing step,

$$\begin{aligned} v_1(j) &= \pi_j b_j(o_1) \quad 1 \le j \le N \\ bt_1(j) &= 0 \qquad\qquad 1 \le j \le N \end{aligned} \tag{4.13}$$

Recursion step,

$$\begin{aligned} v_t(j) &= \max_{i=1}^{N} v_{t-1}(i) a_{ij} b_j(o_t); \quad 1 \le j \le N, 1 < t \le T \\ bt_t(j) &= \operatorname{argmax} v_{t-1}(i) a_{ij} b_j(o_t); \quad 1 \le j \le N, 1 < t \le T \end{aligned} \tag{4.14}$$

Terminating step,

$$\begin{aligned} &\text{The best score:} \quad P* = \max_{i=1}^{N} v_T(i) \\ &\text{The start of backtrace:} \quad q_{T*} = \operatorname*{argmax}_{i=1} v_T(i) \end{aligned} \tag{4.15}$$

The learning problem is where the parameters for an HMM model are learned. If we are given an observation sequence and the possible states, we have to find the A and the B of an HMM model, $\lambda = (A, B)$. The training uses an expectation-maximization algorithm named the "Baum-Welch" algorithm. The transitional probabilities and the emission probabilities are trained in the algorithm. The steps for this algorithm are similar to the forward algorithm but contain additional expectation and maximization steps.

Expectation step,

$$\begin{aligned} \gamma_t(j) &= \frac{\alpha_t(j)\beta_t(j)}{\alpha_T(q_F)} \forall t \text{ and } j \\ \xi_t(i,j) &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\alpha_T(q_F)} \forall t, i, \text{ and} \end{aligned} \tag{4.16}$$

Here, γ denotes the expected state occupancy count,
ξ indicates the expected state transition count

Maximization step,

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \sum_{k=1}^{N} \xi_t(i,k)}$$
$$\hat{b}_j(v_k) = \frac{\sum_{t=1 s.t. O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{4.17}$$

Here, $\hat{a}_{ij}$ is the total expected number of transitions from state i
and $\hat{b}_j(v_k)$ is the observation probability

4.3.3 Implementation of Hidden Markov Model

Our primary purpose is to validate the observation sequence. So, we have to use the decoding problem of the HMM as we will be predicting the states from the observation sequence (sensor values for "Fever"). There is a framework to implement the HMM algorithm. So, to start, we have installed the "hmmlearn" library. After successfully installing, we imported the "hmm" function from there. We declared the transitional probability, emission probability, states, and observations. Then we created a "MultinomialHMM" instance. As the "n_componenets" parameters, we have set it to 2 as we have only two states of fever, either the target person has a fever(1) or not(0). Also, we have set all the required probabilities in the model. After fitting the model with observed fever readings, we were provided with a validated data for the sensor values of the "Fever" attribute. Same process is used for validating the observation sequence of the breathing difficulty. The person will have two states in this case too, breathing difficulty(1) or no breathing difficulty(0).

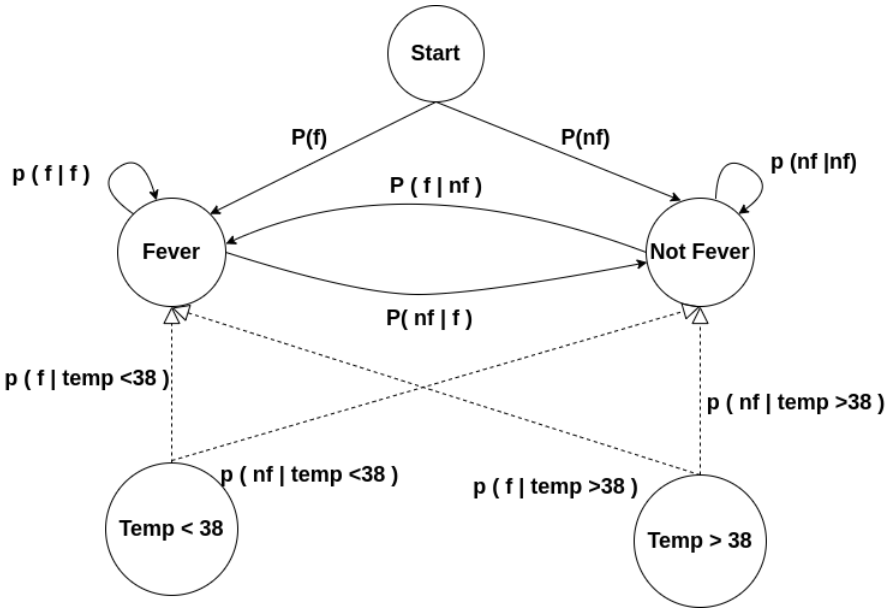


Figure 4.4: Hidden Markov Model for Fever Validation

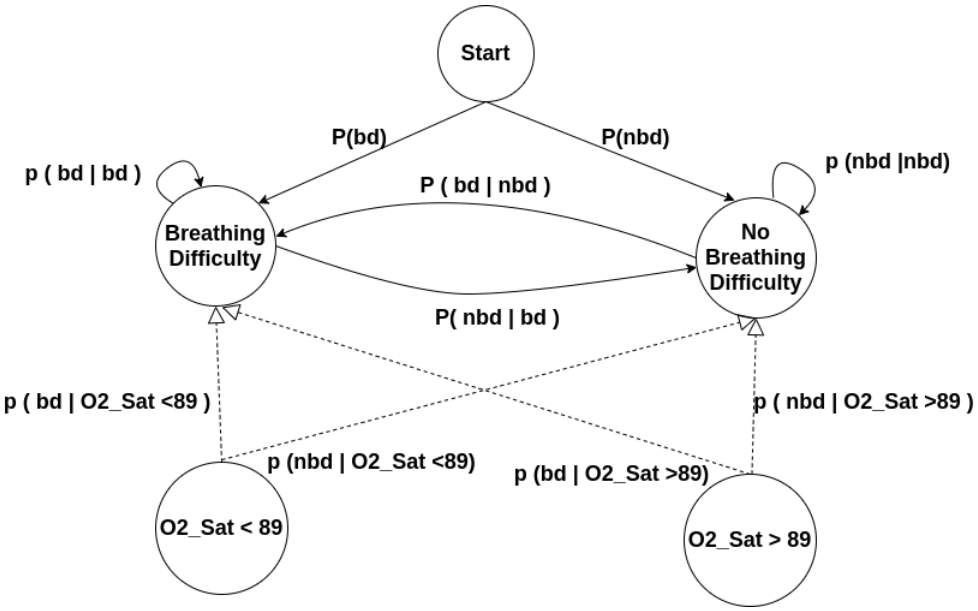


Figure 4.5: Hidden Markov Model for Breathing Difficulty Validation

Chapter 5

Conclusion

5.1 Limitations

Our project has some limitations which was the direct result of the challenges we faced during the research. Here we are discussing those limitations

- Our model is not trained on real data rather we used a synthesized data based on WHO guidelines. If we used a real life data then our model would have been more realistic.
- For Hidden Markov Model, we also could not find the dataset. As a result we needed to use assumptions which will also hinder the accuracy of the model.
- We could not build a wearable system rather we made a prototype of it which is not wearable. If an wearable system could be built then we could get data from the patients.

5.2 Future Works

There are some works that still needs to be done in order to fine-tune the framework. These are the future works that we have decided we will do

- Collect real life data from the COVID-19 patients in order to make the models more accurate.
- Make an integrated wearable system which can be easily wore.
- Select the CPU based on the requirement of the system rather than a generalized one for better performance.
- Select a better audio recording protocol for minimum audio loss.
- Incorporate any symptoms to the model and framework which might be seen in the future COVID-19 patients.

In this paper we have proposed an IoT based Ambient Assisted Living framework for prediction of COVID-19 in elderly people using Deep Neural Networks. The goal of this paper was to develop an 24/7 autonomous automated monitoring environment without hindering the normal activities of the user and informing the user to perform COVID-19 test if there were any anomalous behaviour. Inorder to monitor the users behaviour multiple wearable sensors and non-wearable sensors were placed in the users body and the environment in which the user resides . We have incorporated both respiratory, gastrointestinal and several key symptoms that are being present in COVID-19 patients as the COVID-19 virus is mutating, some common symptoms that are known may not occur as a result when a patient is diagnosed with COVID-19 it already may be too late. We have used HMM to validate some sensors reading where there could be fluctuations. This framework would help the elderly people from visiting the COVID-19 testing center multiple times where they might be healthy but could contact the virus while coming back home. Although we were practically unable to collect data due to the pandemic situation, we have achieved 94.82% accuracy using Deep Neural Networks on a Kaggle Dataset synthesized using WHO guidelines. Elderly people are the primary victims of this virus and if our framework could be implemented practically a number of lives could be saved as the number of waves of this virus is still not known.

Bibliography

[1] D. Monekosso, F. Florez-Revuelta, and P. Remagnino, "Ambient assisted living [guest editors' introduction]," *IEEE Intelligent Systems*, vol. 30, no. 4, pp. 2–6, 2015.

[2] I. Services, *Ambient assisted living for resident care - what is it?* Sep. 2020. [Online]. Available: https://syno.global/ambient-assisted-living/#:~:text=Ambient%20Assisted%20Living%20(AAL)%20is,as%20possible,%20without%20intrusive%20behaviours.

[3] *Introduction to pervasive computing*, Feb. 2020. [Online]. Available: https://www.geeksforgeeks.org/introduction-to-pervasive-computing/.

[4] *What is the internet of things (iot)?* [Online]. Available: https://www.oracle.com/internet-of-things/what-is-iot/.

[5] W. H. Organization *et al.*, "Coronavirus disease (covid-19)," 2020.

[6] J. Ducharme, *The who just declared coronavirus covid-19 a pandemic*, Mar. 2020. [Online]. Available: https://time.com/5791661/who-coronavirus-pandemic-declaration/.

[7] J. Gallagher, *Coronavirus: What it does to the body*, Mar. 2020. [Online]. Available: https://www.bbc.com/news/health-51214864.

[8] M. Cascella, M. Rajnik, A. Aleem, S. C. Dulebohn, and R. D. Napoli, *Features, evaluation, and treatment of coronavirus (covid-19)*, Apr. 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK554776/.

[9] D. R. w. RFI, *Research shows young people are 50% less likely to catch covid-19*, Jun. 2020. [Online]. Available: https://www.rfi.fr/en/science-and-technology/20200616-research-shows-young-people-are-50-less-likely-to-catch-covid-19.

[10] R. Nania, *95 percent of americans killed by covid were 50*, Oct. 2020. [Online]. Available: https://www.aarp.org/health/conditions-treatments/info-2020/coronavirus-deaths-older-adults.html.

[11] T. M. McMichael, D. W. Currie, S. Clark, S. Pogosjans, M. Kay, N. G. Schwartz, J. Lewis, A. Baer, V. Kawakami, M. D. Lukoff, *et al.*, "Epidemiology of covid-19 in a long-term care facility in king county, washington," *New England Journal of Medicine*, vol. 382, no. 21, pp. 2005–2011, 2020.

[12] Y. Zoabi, S. Deri-Rozov, and N. Shomron, "Machine learning-based prediction of covid-19 diagnosis based on symptoms," *npj digital medicine*, vol. 4, no. 1, pp. 1–5, 2021.

[13] G. K. .-J. 18, G. K. -, A. K. S. C. Developer, Marketer, G. K. S. C. Developer, Marketer, G. W. writers are IoT experts, enthusiasts interested in sharing their insights with the IoT industry through IoT For All. If you're interested in contributing to IoT For All, G. writers are IoT experts, and enthusiasts interested in sharing their insights with the IoT industry through IoT For All. If you're interested in contributing to IoT For All, *Where do wearables fit into the internet of things?* Jul. 2020. [Online]. Available: https://www.iotforall.com/where-wearables-fit-in-iot.

[14] A. Dohr, R. Modre-Opsrian, M. Drobics, D. Hayn, and G. Schreier, "The internet of things for ambient assisted living," in *2010 seventh international conference on information technology: new generations*, Ieee, 2010, pp. 804–809.

[15] A. J. Jara, M. A. Zamora-Izquierdo, and A. F. Gómez-Skarmeta, "An ambient assisted living system for telemedicine with detection of symptoms," in *International Work-Conference on the Interplay Between Natural and Artificial Computation*, Springer, 2009, pp. 75–84.

[16] M. Otoom, N. Otoum, M. A. Alzubaidi, Y. Etoom, and R. Banihani, "An iot-based framework for early identification and monitoring of covid-19 cases," *Biomedical Signal Processing and Control*, vol. 62, p. 102 149, 2020.

[17] T. M. McMichael, S. Clark, S. Pogosjans, M. Kay, J. Lewis, A. Baer, V. Kawakami, M. D. Lukoff, J. Ferro, C. Brostrom-Smith, *et al.*, "Covid-19 in a long-term care facility—king county, washington, february 27–march 9, 2020," *Morbidity and Mortality Weekly Report*, vol. 69, no. 12, p. 339, 2020.

[18] O. Taiwo and A. E. Ezugwu, "Smart healthcare support for remote patient monitoring during covid-19 quarantine," *Informatics in Medicine Unlocked*, vol. 20, p. 100 428, 2020.

[19] S. D. Achirei, O. Zvoristeanu, A. Alexandrescu, N. A. Botezatu, A. Stan, C. Rotariu, R. G. Lupu, and S. Caraiman, "Smartcare: On the design of an iot based solution for assisted living," in *2020 International Conference on e-Health and Bioengineering (EHB)*, IEEE, 2020, pp. 1–4.

[20] T. Manoj and G. Thyagaraju, "Ambient assisted living: A research on human activity recognition and vital health sign monitoring using deep learning approaches,"

[21] J. S. Obeid, M. Davis, M. Turner, S. M. Meystre, P. M. Heider, E. C. O'Bryan, and L. A. Lenert, "An artificial intelligence approach to covid-19 infection risk assessment in virtual visits: A case report," *Journal of the American Medical Informatics Association*, vol. 27, no. 8, pp. 1321–1325, 2020.

[22] R. Ye, X. Zhou, F. Shao, L. Xiong, J. Hong, H. Huang, W. Tong, J. Wang, S. Chen, A. Cui, *et al.*, "Feasibility of a 5g-based robot-assisted remote ultrasound system for cardiopulmonary assessment of patients with coronavirus disease 2019," *Chest*, vol. 159, no. 1, pp. 270–281, 2021.

[23] N. K. Suryadevara and S. C. Mukhopadhyay, "Determining wellness through an ambient assisted living environment," *IEEE Intelligent Systems*, vol. 29, no. 3, pp. 30–37, 2014.

[24] *Symptoms of covid-19*. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html.

[25] J. Zhang, "Development of a non-contact infrared thermometer," in *2017 International Conference Advanced Engineering and Technology Research (AETR 2017)*, Atlantis Press, 2018, pp. 308–312.

[26] R. B. Kaplan, T. M. Johnson, R. Schneider, and S. M. Krishnan, "A design for low cost and scalable non-contact fever screening system," in *ASEE Annual Conference and Exposition, Conference Proceedings*, 2011.

[27] D. J. Diaz, *Episode 33 - medical oxygen*, Apr. 2021. [Online]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/media-resources/science-in-5/episode-33---medical-oxygen.

[28] J. Wan, Y. Zou, Y. Li, and J. Wang, "Reflective type blood oxygen saturation detection system based on max30100," in *2017 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, IEEE, 2017, pp. 615–619.

[29] E. Suprayitno, M. Marlianto, and M. Mauliana, "Measurement device for detecting oxygen saturation in blood, heart rate, and temperature of human body," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1402, 2019, p. 033 110.

[30] B. Nazario, *Symptoms of coronavirus: Early signs, serious symptoms and more*, Feb. 2021. [Online]. Available: https://www.webmd.com/lung/covid-19-symptoms#1.

[31] L. Orlandic, T. Teijeiro, and D. Atienza, "The coughvid crowdsourcing dataset: A corpus for the study of large-scale cough analysis algorithms," *arXiv preprint arXiv:2009.11644*, 2020.

[32] D. J. S. Schulman, *Is a sore throat a typical symptom of covid-19?* Jul. 2020. [Online]. Available: https://www.healthline.com/health/sore-throat-coronavirus.

[33] L.-M. Weng, X. Su, and X.-Q. Wang, "Pain symptoms in patients with coronavirus disease (covid-19): A literature review," *Journal of Pain Research*, vol. 14, p. 147, 2021.

[34] *Is a runny nose a symptom of covid-19?* Apr. 2021. [Online]. Available: https://covid.joinzoe.com/post/covid-runny-nose.

[35] R. Rettner, *Diarrhea is first sign of illness for some covid-19 patients*, Mar. 2020. [Online]. Available: https://www.livescience.com/coronavirus-diarrhea-symptoms.html.

[36] C. Han, C. Duan, S. Zhang, B. Spiegel, H. Shi, W. Wang, L. Zhang, R. Lin, J. Liu, Z. Ding, *et al.*, "Digestive symptoms in covid-19 patients with mild disease severity: Clinical presentation, stool viral rna testing, and outcomes," *The American journal of gastroenterology*, 2020.

[37] L. L. Maragakis, *Coronavirus symptoms: Frequently asked questions*, Feb. 2021. [Online]. Available: https://www.hopkinsmedicine.org/health/conditions-and-diseases/coronavirus/coronavirus-symptoms-frequently-asked-questions.

[38] H. Hari, *Symptoms and covid presence*, Aug. 2020. [Online]. Available: https://www.kaggle.com/hemanthhari/symptoms-and-covid-presence.

[39] A. Thanda, A. T. A. T. O. from India, A. T. A. Thanda, A. Thanda, and O. f. India, *What is logistic regression? a beginner"s guide [2021]*, May 2021. [Online]. Available: https://careerfoundry.com/en/blog/data-analytics/what-is-logistic-regression/.

[40] S. Swaminathan, *Logistic regression - detailed overview*, Jan. 2019. [Online]. Available: https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc.

[41] J. Brownlee, *Logistic regression for machine learning*, Aug. 2020. [Online]. Available: https://machinelearningmastery.com/logistic-regression-for-machine-learning/.

[42] N. S. Chauhan, *Decision tree algorithm, explained*, Jan. 2020. [Online]. Available: https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html.

[43] N. Donges, *A complete guide to the random forest algorithm*, Jun. 2019. [Online]. Available: https://builtin.com/data-science/random-forest-algorithm.

[44] M. Schott, *Random forest algorithm for machine learning*, Feb. 2020. [Online]. Available: https://medium.com/capital-one-tech/random-forest-algorithm-for-machine-learning-c4b2c8cc9feb.

[45] M. McGregor, *Svm machine learning tutorial – what is the support vector machine algorithm, explained with code examples*, Jul. 2020. [Online]. Available: https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-algorithm-explained-with-code-examples/.

[46] S. Awasthi, *Seven most popular svm kernels*, Dec. 2020. [Online]. Available: https://dataaspirant.com/svm-kernels/.

[47] M. Pathak, *(tutorial) learn to use xgboost in python*, Nov. 2019. [Online]. Available: https://www.datacamp.com/community/tutorials/xgboost-in-python.

[48] V. Morde, *Xgboost algorithm: Long may she reign!* Apr. 2019. [Online]. Available: https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d.

[49] S. Chen, *Simple math about xgboost*, Jul. 2020. [Online]. Available: https://towardsdatascience.com/simple-math-about-xgboost-b9a924aaae9f.

[50] A. Samudrala, *Unveiling mathematics behind xgboost*, Aug. 2018. [Online]. Available: https://www.kdnuggets.com/2018/08/unveiling-mathematics-behind-xgboost.html.

[51] J. J. Moolayil, *A layman's guide to deep neural networks*, May 2020. [Online]. Available: https://towardsdatascience.com/a-laymans-guide-to-deep-neural-networks-ddcea24847fb.

[52] S. Raj, *How to evaluate the performance of your machine learning model*, Sep. 2020. [Online]. Available: https://www.kdnuggets.com/2020/09/performance-machine-learning-model.html.

[53] J. Frost, "Measures of central tendency: Mean, median, and mode," [Online]. Available: https://statisticsbyjim.com/basics/measures-central-tendency-mean-median-mode/.

[54] R. Nall, *Body temperature: Normal ranges in adults and children*, Jan. 2020. [Online]. Available: https://www.medicalnewstoday.com/articles/323819#adults.

[55] *Pulse oximetry test: Uses, procedure, risks, results.* [Online]. Available: https://www.webmd.com/lung/pulse-oximetry-test.

[56] E. Suni, *How much sleep do we really need?* Mar. 2021. [Online]. Available: https://www.sleepfoundation.org/how-sleep-works/how-much-sleep-do-we-really-need.

[57] *How often should i go to the bathroom?* Sep. 2016. [Online]. Available: https://share.upmc.com/2016/09/healthy-bowel-movements/#:~:text=Unlike%20constipation,%20diarrhea%20happens%20when,stool%20is%20loose%20and%20watery..

[58] D. H. Jurafsky and J. H. Martin. [Online]. Available: https://web.stanford.edu/~jurafsky/slp3/A.pdf.