A

THESIS

On

"SIGN LANGUAGE RECOGNITION SYSTEM USING ML"

Project Report Submitted to

CHHATTISGARH SWAMI VIVEKANAND TECHNICAL UNIVERSITY BHILAI (INDIA)

For the partial fulfilment of the requirement for the award of Degree of

Bachelor of Engineering

In

Electrical & Electronics Engineering

By

ANKIT KUMAR
MOHIT KUMAR GUPTA
YANENDRA KUMAR

Under the Guidance of

Miss. Poorva Sharma

Department of Electrical & Electronics Engineering
Government Engineering College Raipur
Old Dhamtari Road Sejbahar Raipur Chhhattisgarh India
-492015 Phone- 0771-2103157, Website - www.gecraipur.ac.in
Email- principal@gecraipur.ac.in

SESSION: 2018-2022

DECLARATION

We, the undersigned, solemnly declare that the report of thesis work entitled **“SIGN LANGUAGE RECOGNITION SYSTEM USING ML”** is based on work carried out during the course of study under the supervision of **MISS. POORVA SHARMA** Assistant Professor, Department of Electrical & Electronics Engineering, Government Engineering College Raipur, Chhattisgarh.

We assert that the statements made & conclusions drawn are an outcome of the project work. We further declare that to the best of our knowledge & belief that the report does not contain any part of any work submitted for the award of B.E. Degree in this University.

We further certify that

i. The work contained in the report is original and has been done by us under the general supervision of our supervisor(s).

ii. The work has not been submitted to any other Institute for any other degree/diploma/certificate in this University or the any other University of India or abroad.

iii. We have followed the guidelines provided by the University in writing the report.

iv. Whenever we have used materials (data, theoretical analysis, and text) from other sources, we have given due credit to them by citing them in the text of the report and giving their

S.No.	Name of Student	Roll Number	Enrolment Number	Signature
1.	**ANKIT KUMAR**	**301602518011**	**BF3200**	
2.	**MOHIT KUMAR GUPTA**	**301602518034**	**BF3223**	
3.	**YANENDRA KUMAR**	**301602518049**	**BF3238**	

CERTIFICATE

This is to certify that the work incorporated in this thesis is entitled **"SIGN LANGUAGE RECOGNITION SYSTEM USING ML."**

S.No.	Name of Student	Roll Number	Enrollment Number	Signature
1.	**ANKIT KUMAR**	**301602518011**	**BF3200**	
2.	**MOHIT KUMAR GUPTA**	**301602518034**	**BF3223**	
3.	**YANENDRA KUMAR**	**301602518049**	**BF3238**	

Is the report of project work carried out under my guidance and supervision for an award of the Degree **Bachelor of Engineering** in the student of Electrical & Electronics Engineering of Chhattisgarh Swami Vivekananda Technical University, Bhilai (C.G.), India.

To the best of my knowledge and belief, the thesis

i. Embodies the work of the candidate herself.

ii. Has duly been completed.

iii. Fulfill the requirement of the ordinance relating to BE Degree of the University.

iv. It is up to the desired standard both in respect of contents and language for being referred to the examiners.

……………………….

(Signature of the Guide)
MISS. POORVA SHARMA

……………………….

(Signature of the Head of Department with seal)
Dr. R.S. Parihar
HOD

Forwarded to Chhattisgarh Swami Vivekanand Technical University, Bhilai

CERTIFICATE BY THE EXAMINERS

This is to certify that the project work entitled **"SIGN LANGUAGE RECOGNITION SYSTEM USING ML"** submitted by

S.NO	STUDENT NAME	ROLL NUMBER	ENROLMENT NUMBER	SIGNATURE
1	**ANKIT KUMAR**	**301602518011**	**BF3200**	
2	**MOHIT KUMAR GUPTA**	**301602518034**	**BF3223**	
3	**YANENDRA KUMAR**	**301602518049**	**BF3238**	

Under the Supervision of **MISS. POORVA SHARMA** HOD, Department of Electrical & Electronics Engineering, Government Engineering College Raipur, Chhattisgarh hereby accepted and approved for submission to Chhattisgarh Swami Vivekanand Technical University, Bhilai (C.G.) in the partial fulfilment of the requirement for the award of the Degree of Bachelor of Engineering in Electrical & Electronics Engineering.

It is understood that by this approval, the undersigned does not endorse or approve any statement made, opinion expressed, or conclusion drawn therein but only approves the report for the purpose for which it is being submitted.

______________________ ______________________

(Signature of the External Examiner) (Signature of the Internal Examiner)

(Name of the External Examiner) (Name of the Internal Examiner)

Date: Date:

Designation: Designation:

Institute: Institute:

ACKNOWLEDGEMENT

We sincerely acknowledge with deep gratitude the valuable guidance renewed from **MISS. POORVA SHARMA** of Government Engineering College, Raipur, Chhattisgarh. He was a constant source of imputation during our project and provided his expert and sagacious guidance to end unrestrained cooperation. He has given us generous encouragement. We are presenting this project entitled **“SIGN LANGUAGE RECOGNITION SYSTEM USING ML.”**

We are highly thankful to **MISS. POORVA SHARMA** Head of Electrical & Electronics Engineering Department, for providing us with necessary facilities and cooperation during the course of study

We would also like to avail this opportunity to express our sense of gratitude toward the principal of our institute for their inspiration and fruitful advice, and active support.

There are many people who have helped and supported us, and we would like to take this opportunity to thank everyone.

S.No.	Name of Student	Roll Number	Enrolment Number	Signature
1.	**ANKIT KUMAR**	**301602518011**	**BF3200**	
2.	**MOHIT KUMAR GUPTA**	**301602518034**	**BF3223**	
3.	**YANENDRA KUMAR**	**301602518049**	**BF3238**	

# ABSTRACT

Our group designed and built a glove that can help a person learn gestures To alphabets in sign language. The device consists of a sensing unit that detects the gesture and a classification algorithm to check each motion. This device is connected to the computer Via Arduino with Bluetooth connectivity to avoid any wire, making the device user-friendly.

Salient Points-

•The project uses a sensor called flex that measures the blend of the finger. Five flex has to be attached to the glove to capture the signs (American sign language) performed by the user and transfer them to the processing unit (Arduino Nano BLE) that translates/convert them into understandable English sentences.

•The glove can be used by any deaf/mute person to communicate with people who do not understand sign language.

•The performed signs with the hand wearing the gloves are displayed as alphabetical letters on the computer display.

• These gloves can be handy, efficient, and helpful for solving the communication barrier between ordinary and physically disabled people. It is also beneficial for the application of machine learning.

-The glove output is limited to a single hand, and since hands differ in size and freedom of motion, therefore work can be done on this glove so that the wearable hand glove mouse can be flexible for every hand size.

CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

This project aims to lower this barrier in communication. It is based on the need to develop an electronic device that can translate sign language into speech in order to make the communication take place between the mute communities with the general public possible.

A Wireless data glove is used, which is a normal cloth driving glove fitted with flex sensors along the length of each finger. Mute people can use gloves to perform hand gestures, and they will be converted into speech so that normal people can understand their expressions. A gesture in sign language is a particular movement of the hands with a specific shape made out of them.

A **sign language glove** is an electronic device that attempts to convert the motions of a sign language into written or spoken words. Some critics of such technologies have argued that the potential of sensor-enabled gloves to do this is commonly overstated or misunderstood because many sign languages have a complex grammar that includes the use of the sign space and facial expressions (non-manual elements).

Sign language is a language that is used by deaf and dumb people. To communicate with a normal person using this language, this smart glove is used. Here the glove is fixed with the microcontroller and flex sensors which are further powered by a lipo battery. When a person uses this smart glove to communicate with the normal person, the flex sensors detect the gestures and send them to the Arduino BLE where the process takes place, and the output is shown in the form of text and audio to the normal person. Thus the communication for deaf/dumb people becomes easy using this method.

1.1HISTORY

History - The primary idea dates to the 1980s, when researchers started exploring how humans could interact with computers using gestures. In 1983, a Bell Labs engineer named Gary Grimes invented a glove for data entry using the 26 manual gestures of the American Manual Alphabet, used by speakers of American Sign Language.

1.2 IDEA BEHIND THIS PROJECT

1.2.1 Problem statements-

Case I - Consider two Especially Abled People. One person is Blind, and the other person is Dumb, so if they want to communicate with each other, they have a problem doing that. Also, when a Dumb person uses sign language to communicate with a blind person, a blind person cannot understand the sign language. Generally, deaf and dumb people use sign language for communication. Still, they find it challenging to communicate with others who don't understand sign language. Sign language is an expressive and natural way to communicate between ordinary and dumb people (mainly through hand gestures).

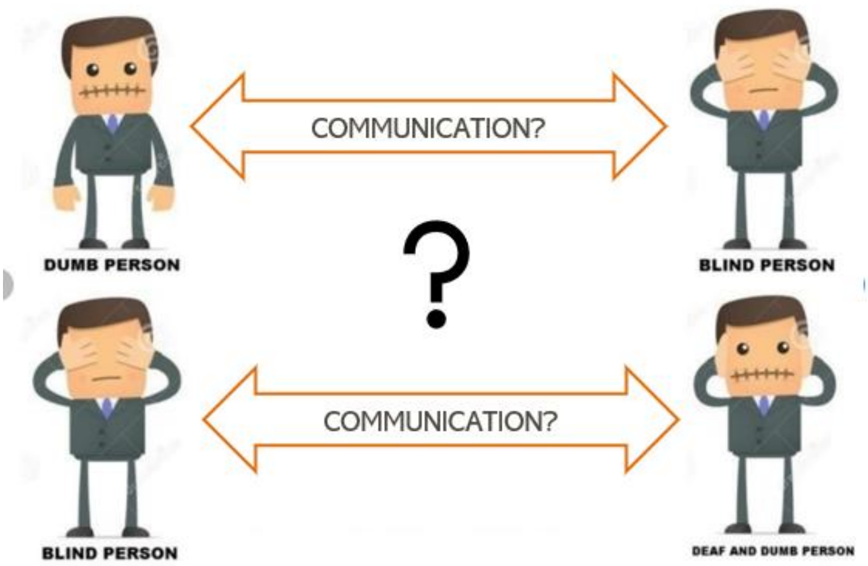

Figure 1.1 communication barrier

Case-ii -Consider two people who want to comminate. One person is average (with no physical disability). Another one is DUMB, so now what happens is that Generally deaf and dumb people use sign language for communication. Still, they find it challenging to communicate with others who don't understand sign language. Sign language is an expressive and natural way to communicate between normal and dumb people (information is mostly conveyed through hand gestures).

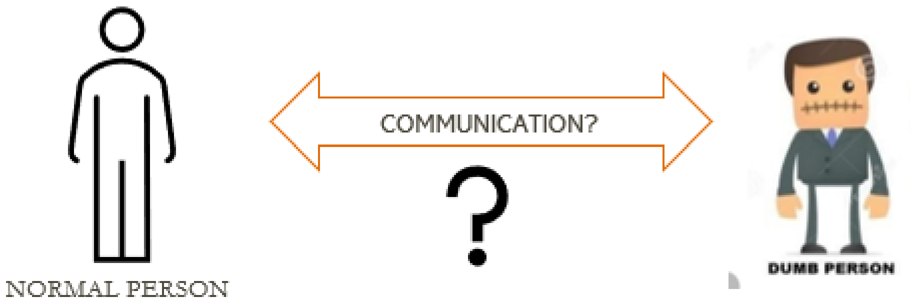

Figure.1.2 communication barrier

1.2.2 Solutions

As mentioned above, we have to find a way to overcome this problem. We researched this area and found two solutions for this problem. The idea is to create a smart device or system which can translate the sign language into an understandable language such as audio-video.

□The first solution is to make smart gloves that can measure the gesture by using a flex sensor and translating it to the digital data quantity. This can be further processed by using a Microcontroller and P.C.

□The second solution is to make a virtual environment so that the computer can convert sign/gesture language to the desired output. It is basically done by gesture detection using a Camera. and some machine learning process.

1.2.3 Objectives

The Glove Based Sign Language Interpreter forms a bridge of communication for deaf-mute and normal people. It mainly targets the dumb and deaf to be able to communicate their ideas through hand recognition and voice recognition gesture.

Help deaf-mute people integrate into society.

The goal was to create a glove device that detects sign language gestures for letters used in American Sign Language and inputs them into a computer, where a computer program checks the gesture. It can be a good learning tool for sign language, and the idea can be built on in the future.

CHAPTER -2

LITERATURE REVIEW AND THEORY

2.1 LITERATURE SURVEY

Human beings interact with each other to convey their ideas, thoughts, and experiences to the people around them. But this is not the case for deaf-mute people. Sign language paves the way for deaf-mute people to communicate. Through sign language, communication is possible for a deaf-mute person without the means of acoustic sounds. The aim of this work is to develop a system for recognizing sign language, which provides communication between people with speech impairment and normal people, thereby reducing the communication gap between them. Compared to other gestures (arm, face, head, and body), hand gesture plays an important role, as it expresses the user's views in less time. In the current work flex, a sensor-based gesture recognition module is developed. Generally, dumb people use sign language for communication, but they find difficulty in communicating with others who don't understand sign language. This project aims to lower this barrier in communication. It is based on the need to develop an electronic device that can translate sign language into speech in order to make the communication take place between the mute communities with the general public possible. A Wireless data glove is used, which is a normal cloth driving glove fitted with flex sensors along the length of each finger and the thumb. Mute people can use gloves to perform hand gestures, and it will be converted into speech so that normal people can understand their expressions. Sign language is the language used by mute people, and it is a communication skill that uses gestures instead of sound to convey meaning, simultaneously combining hand shapes, orientations, and movement of the hand's arms or body and facial expressions to express a speaker's thoughts fluidly. Signs are used to communicate words and sentences to the audience. This system facilitates communication between silent, hearing-impaired, blind people and normal people. It also helps the mute, hearing-impaired, and blind to interact among themselves. It is not an easy task for normal people to perceive the intended meaning of these sign language used by the hearing-impaired and silent. Moreover, blind people cannot watch their gestures. Official sign language is used by the dumb and deaf but is not familiar with the normal world, and the people who are blind cannot follow sign language. This device converts gestures into voice and vice-versa, which is suitable for both disabled and normal people. To help deaf people, the gestures are converted into text. This text gets displayed on a screen.

2.2 Definitions-

2.2.1 Sign language-

What is sign language?

Sign language is a way of communicating using hand gestures and movements, body language, and facial expressions instead of spoken words. Like any spoken language, such as Italian or Spanish, there are lots of different sign languages across the world. Sign language is used mainly by people who are Deaf or have hearing impairments.

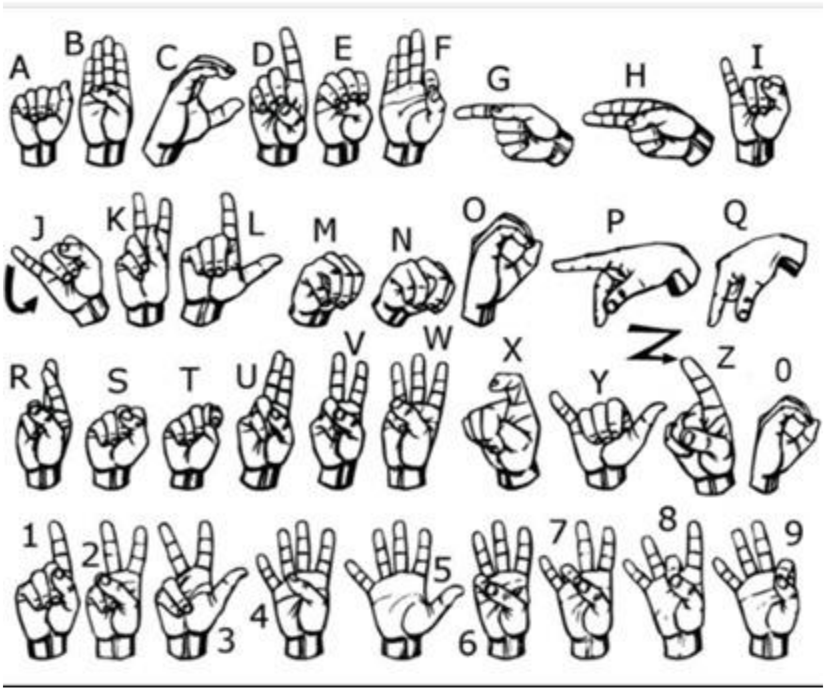

Figure. 2.1 Sign language [17]

2.2.2 Machine learning

Machine learning is **a data analytics technique that teaches computers to do what comes naturally to humans and animals**: learn from experience. Machine learning algorithms use computational methods to “learn” information directly from data without relying on a predetermined equation as a model. Machine learning is a branch of artificial intelligence (A.I.) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. ... Machine learning is an important component of the growing field of data science.

TinyML- Machine learning in embedded systems allows the use of that data in automated business processes to make more educated predictions. Running machine learning models on embedded devices is generally known as embedded machine learning.

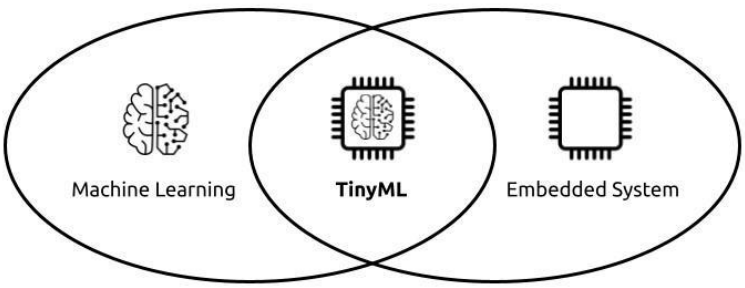

figure 2.2 Embedded ML [18]

2.3. Advantages

- This is a technology of the future that can be of great utility.
- In the future, this project will be useful for deaf and dumb people who cannot communicate with normal people. It is also useful for speech impaired and paralyzed patients means those who do not speak properly.
- This system eliminates the barrier in communication between the mute community and the normal people by discarding the need for an interpreter. It facilitates effective real-time communication.
- It also provides communication between dumb and blind.

2.4 Hardware Requirement

This smart glove represents letters, numbers, and phrases in sign language in the form of hand gestures. We should know the movement in fingers for each number, phrase, and letter. This electronic device has five flex sensors. Each sensor is fixed in fingers (flex sensor has the voltage divider, and the sensor is implemented with resistors.

2.4.1 Flex sensors-

A flex sensor or bend sensor is a sensor that measures the amount of deflection or bending. Usually, the sensor is stuck to the surface, and the resistance of the sensor element is varied by bending the surface. Since the resistance is directly proportional to the amount of bend, it is used as a goniometer and is often called a flexible potentiometer.

Figure 2.3 flex sensor [19]

2.4.2 Arduino Nano 33 BLE

- The Nano 33 BLE Sense is Arduino's 3.3V AI-enabled board.
- Arduino Nano 33 BLE Sense comes with a series of embedded sensors.
- Nine axes inertial sensor: What makes this board ideal for wearable devices
- humidity and temperature sensor: to get a highly accurate measurement of the environmental conditions

- barometric sensor: we could make a simple weather station.
- Microphone: to capture and analyse sound in real-time.
- Gesture, proximity, light colour, and light intensity sensor allows doing projects that need to communicate to other devices at a close range.

Arduino Nano 33 BLE Sense is ideal for interactive automation projects.

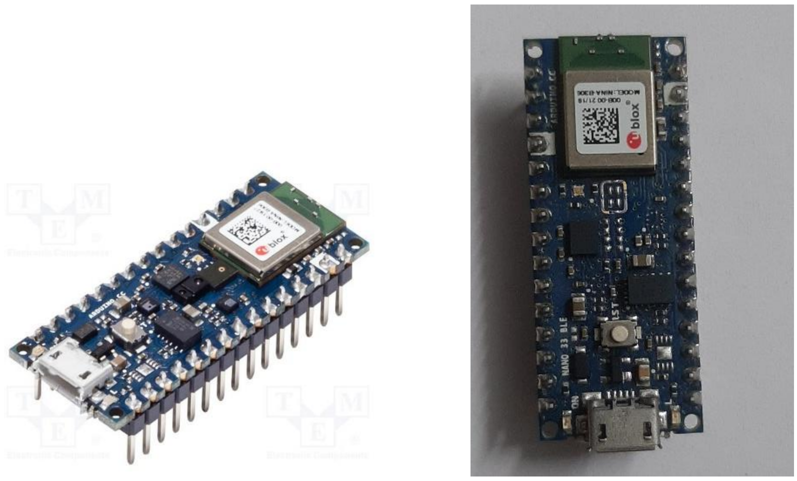

Figure 2.4. Arduino Nano 33 BLE [20]

2.4.3 jumper Wires/Connecting wires-

A jump wire (also known as jumper, jumper wire, jumper cable, DuPont wire, or cable) is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned"), which is normally used **to interconnect the components of a breadboard or other prototype or test**.

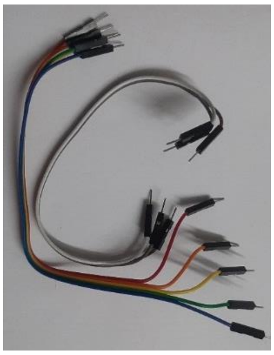

Figure 2.5. i Jumper wire

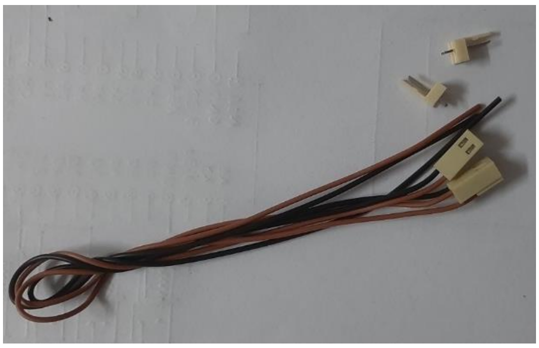

figure 2.5.ii JST 2 PIN

2.4.4 Push-button

A push-button switch controls the action in a machine or other type of process. They are common features within the home and workplace and are also referred to as push-button switches or push switches.

figure.2.6 Pushbuttons

2.4.5 For PCB

2.4.5.1 General purpose PCB

As its name suggests, general-purpose PCBs are **widely used to embed circuits randomly for running hardware**. Its layer is coated with copper and allows proper soldering without any short circuit. General-purpose board connections are not built, so connections are to be created.

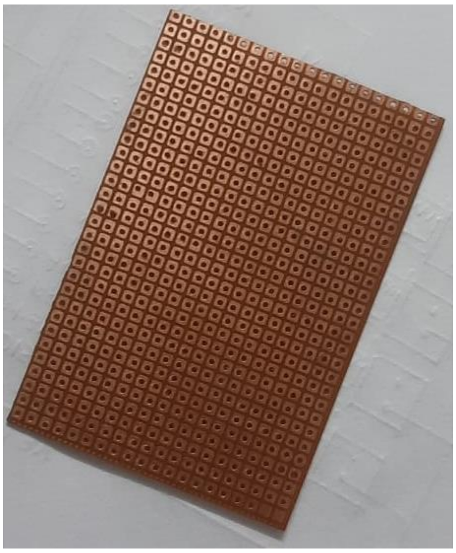

figure 2.7 General Purpose PCB

2.4.5.2 Pin Headers

A pin header is a form of electrical connector. A male pin header consists of one or more rows of metal pins melded into a plastic base, often 2.54 mm apart, though available in many spacings. Male pin headers are cost-effective due to their simplicity.

figure 2.8 Female Pin Headers

2.4.5.3 Heat shrink tube

In general, heat shrink tubing is a common element in most electrical setups that **insulates electrical components from external factors** such as moisture, dust, abrasion, and sharp objects that might otherwise damage wires and electrical components

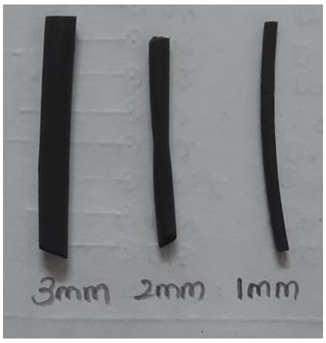

figure 2.9 Heat shrink tube

2.4.6 Hand Gloves-

We need a material, in which we can attach PCB and other sensors. To do so we need hand gloves so that we can easily wear it whenever needed. This glove contains a microcontroller PCB and flex sensors and contact sensors. As shown in figure below – hand gloves

figure 2.10 hand gloves

2.4.7 Resistors-

We are using 10K resistors to connect flex sensors and buttons with Arduino

figure 2.11 Resistors

2.4.8 Power source

We are using a 9v simple general-purpose Battery for this project

Figure 2.12 9v battery

2.5 Software Requirements

2.5.1 Arduino IDE beta Version 2.0

The Arduino Integrated Development Environment is a cross-platform application that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

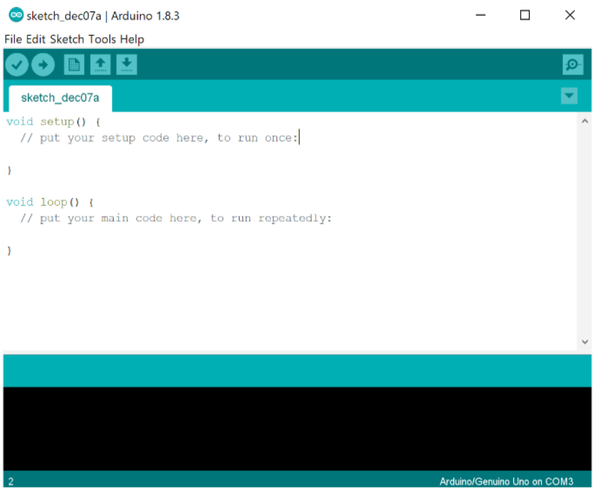

Figure 2.13. Arduino IDE

2.5.2 Tinker-CAD

Tinker cad is a free, easy-to-use web app that equips the next generation of designers and engineers with the foundational skills for innovation: 3D design, electronics, and coding!

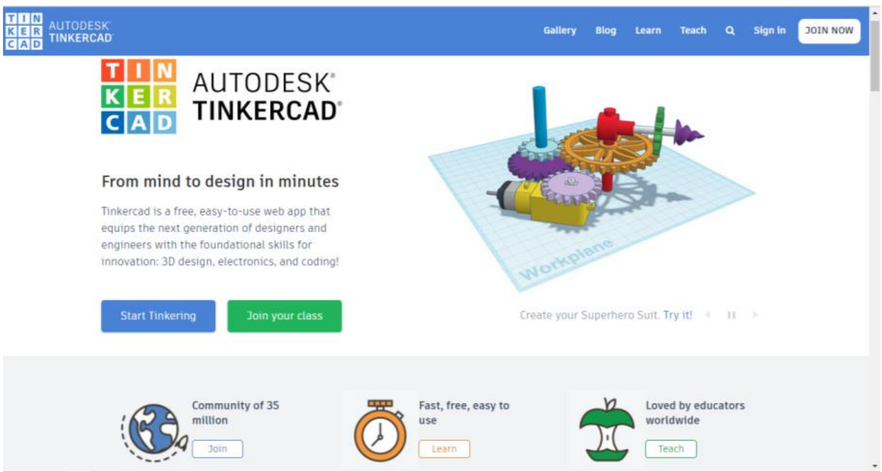

Figure 2.14. tinkerCAD the circuit simulation and 3d model design software [21]

2.5.3 Edge-Impulse

Edge Impulse is the **leading development platform for embedded machine learning**, used by over 1,000 enterprises across 10,000 ML projects worldwide. We are on a mission to enable the ultimate development experience for machine learning on embedded devices for sensors, audio, and computer vision, at scale.

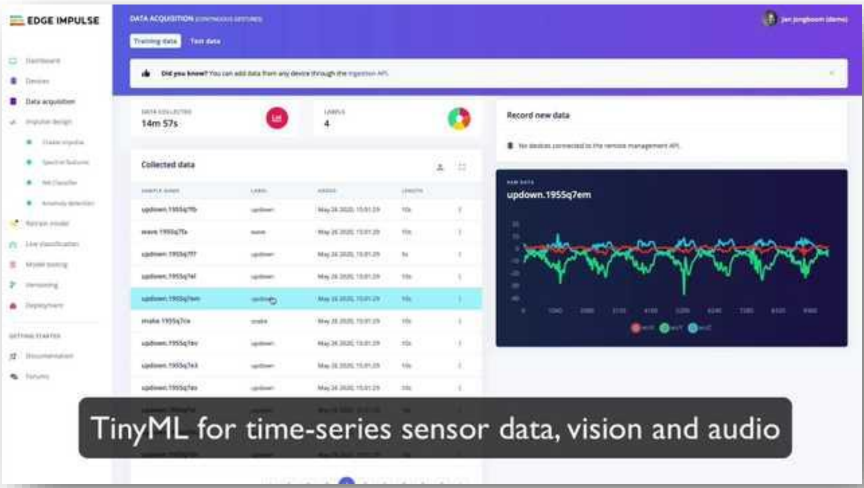

Figure 2.15 Edge impulse- the machine learning platform[11]

2.6 Tools

→ Command Prompt

→Note Pad

→Micro USB cable

CHAPTER -3

METHODOLOGY

3. Assembling Procedure

3.1 Introduction-

We all need to perform a stepwise procedure to achieve destination output.

- STEP 1. COLLECT COMPONENTS
- STEP 2. CIRCUIT DESIGN
- STEP 3. BREADBOARD IMPLEMENTATION
- STEP 4. FINAL CONNECTION WITH PCB
- STEP 5. FINAL SOLID PRODUCT WITHOUT BREADBOARD

3.2 Block diagram-

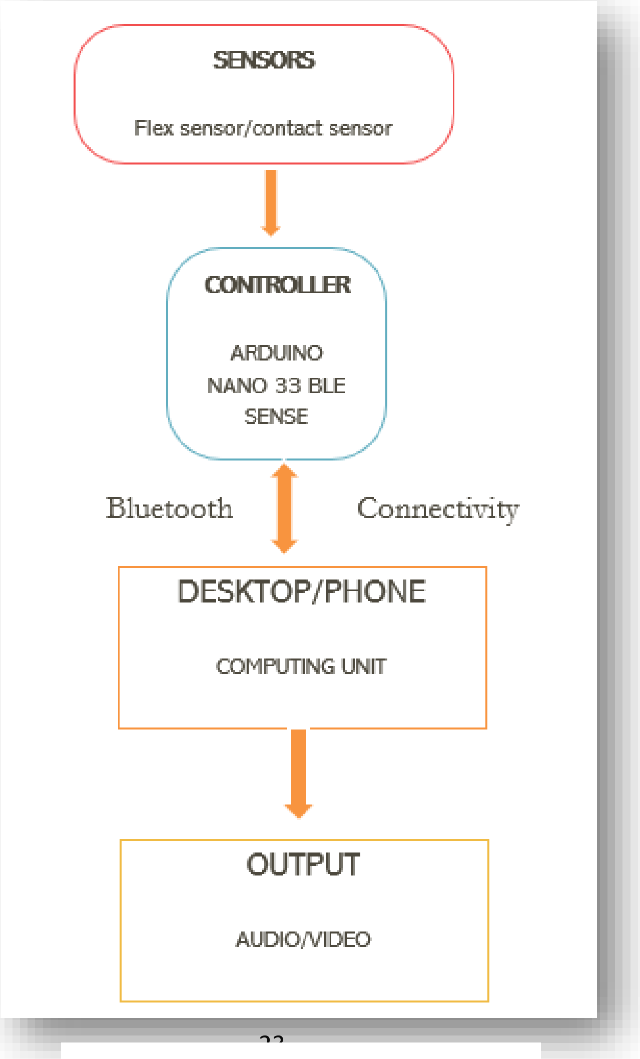

Figure 3.1 Block diagram

3.3 Circuit Diagram-

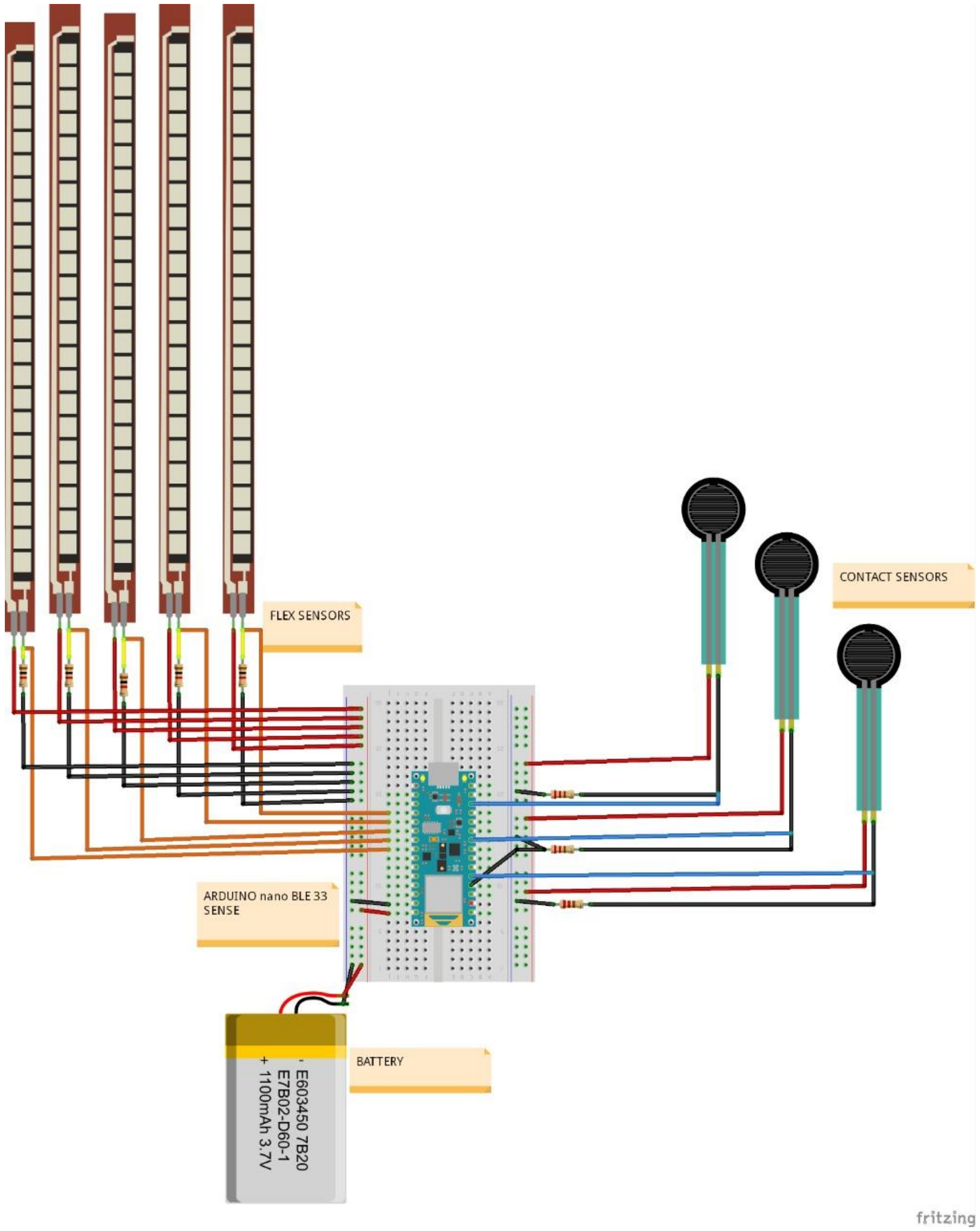

Figure 3.2 Circuit diagram

3.4 Circuit Simulation-

First, we need to simulate whether our circuit is connected correctly or not?

To do so, we are going to implement this circuit in the thinker CAD platform, as shown below

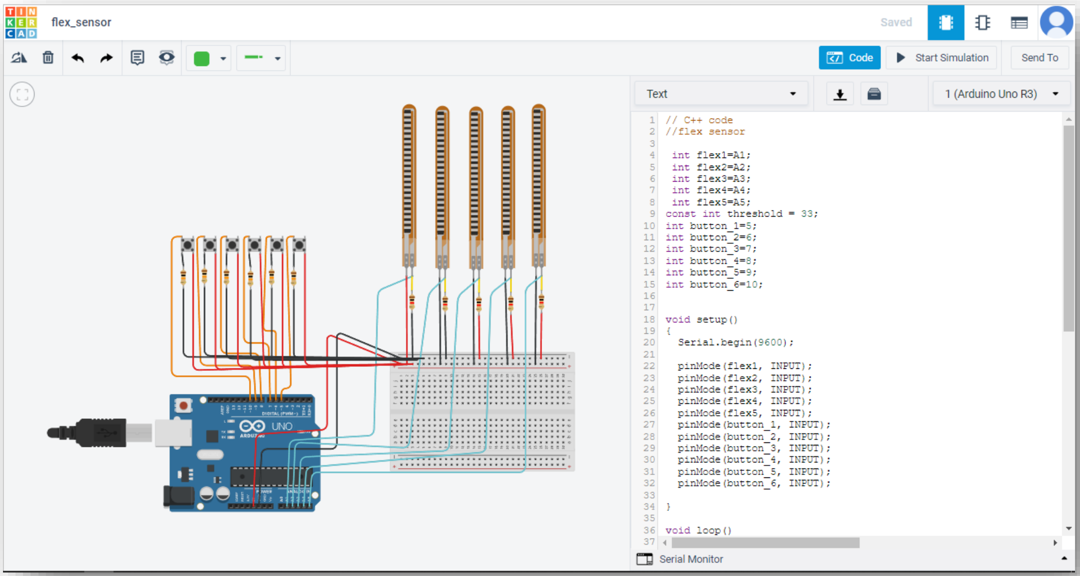

Figure 3.3. simulating circuit

Result- Performing A sign, you can see serial port displaying A text

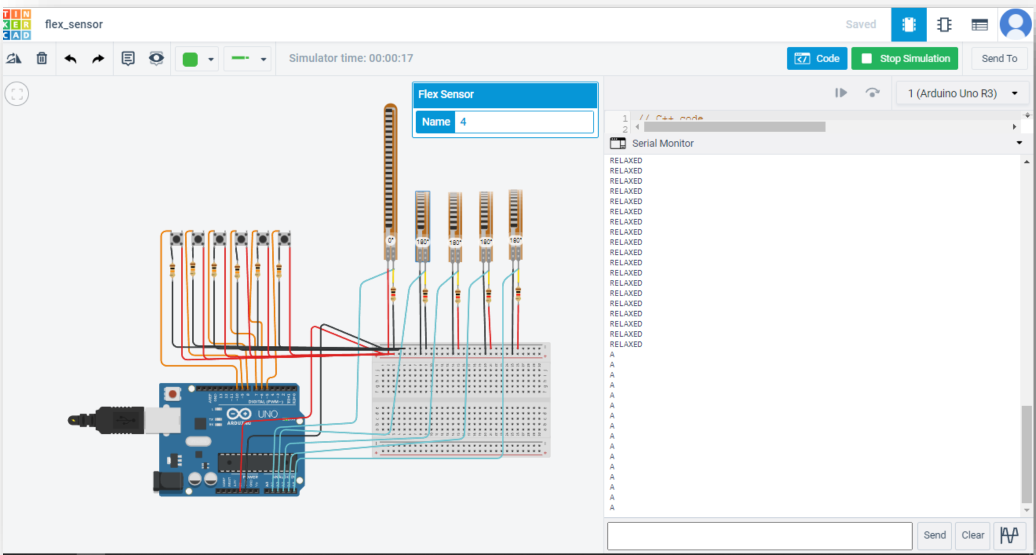

Figure 3.4 simulation result

3.5 3d model-

We've created a prototype 3d model of our project model, i.e., smart gloves, to demonstrate how our product is going to look using tinker cad software.

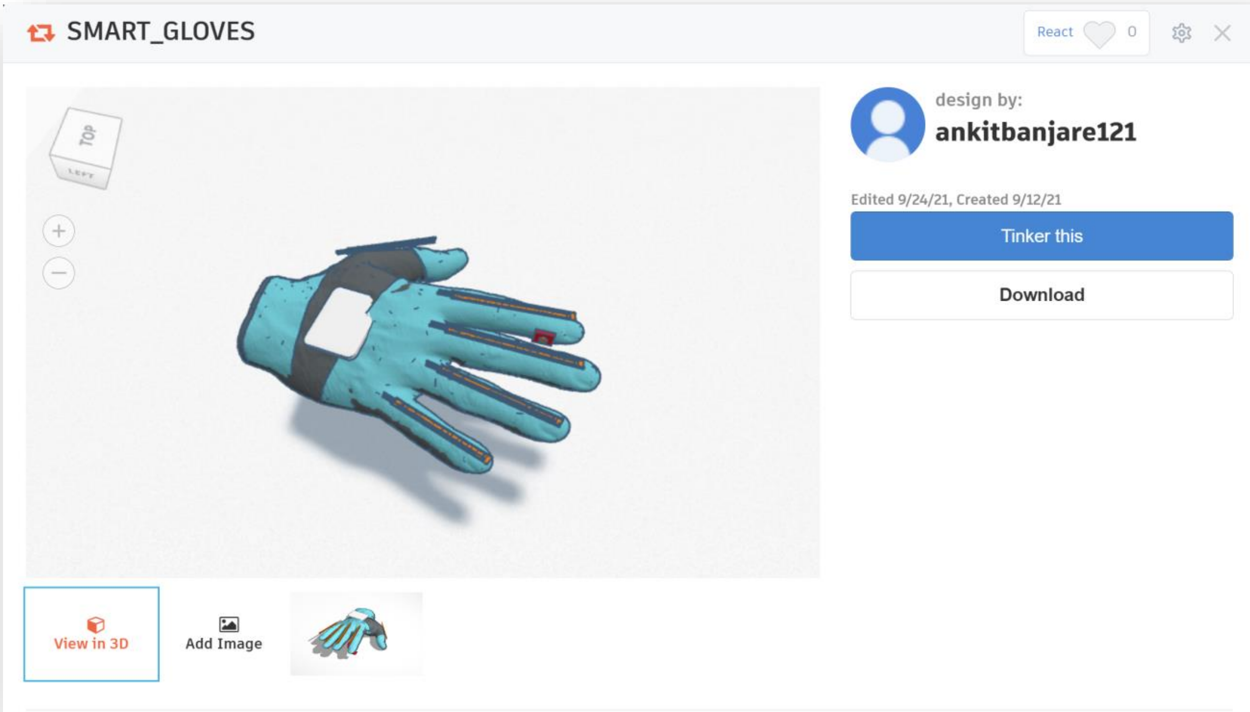

Figure 3.5 3D model of smart gloves

Link-SMART_GLOVES | Tinkercad

3.6 Construction

3.6.1 Workplan

In this project, all operations were performed on Indian Sign Language (ISL). In the ISL manual alphabet, fingerspelling is used primarily for spelling out names or English terms which do not have established signs. The database consists of 26 ISL alphabets others are static gestures. A low-cost hand glove circuit developed with multiple accelerometers is used to capture the hand gestures performed by the performer. It produces the finger flexion of each finger, the movement and orientation of the hand, and the electrical signal from the muscle activities of the hand. The system works online gesture recognition, i.e., the real-time signal from the gloves is given as an input, and the system tells us the matched gesture. It is purely data dependent

3.6.2 Proposed System

Sign language interpreter consisting of accelerometer sensor which helps to measure the movement of the three-axis direction (x, y, and z). The accelerometer sensors are placed on the glove. The data from these sensors are sent to the microcontroller for further processing purposes. Once data is recognized at the microcontroller, that is sent to the android phone via Bluetooth module. On the android phone side, an app is developed, which is text to speech converter that helps to convert text signal to voice signal.

3.6.3 Breadboard implementation

After successfully simulating the circuited model, we need to practically test the circuit connection using a breadboard. As shown figure below-

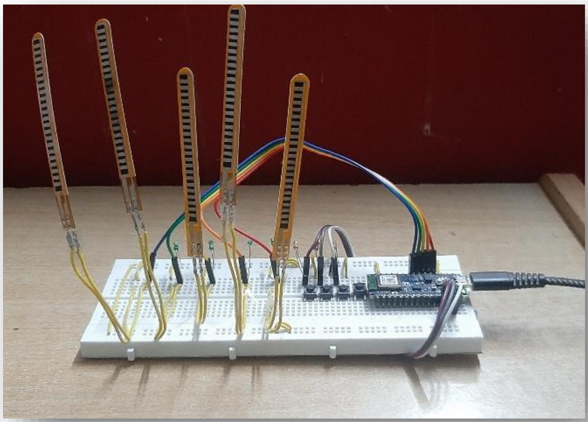

Figure 3.6 breadboard implementation side view

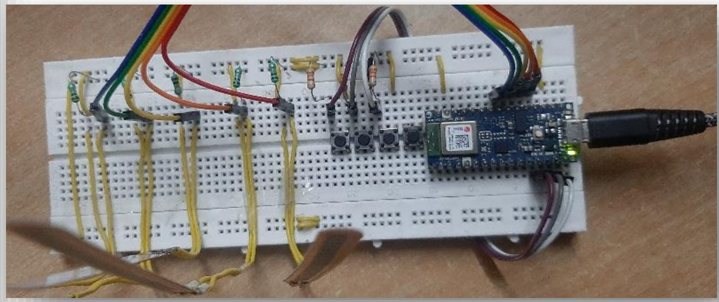

Figure 3.7 breadboard implementation top view

3.6.4 Working

From breadboard Implementation-

- ✓ After connecting all sensors (i.e., five flex sensors, three push-buttons, In-built Gyro Sensor) with embedded microcontroller Arduino nano BLE 33
- ✓ We need to check whether our microcontroller getting all the sensor data or not?
- ✓ By using the edge impulse platform, we can directly plot sensor data.
- ✓ In further titles, we are going to present some demonstrations we've done during testing the breadboard implementation
- ✓ We are also using this edge impulse Platform to train our model (machine Learning part)

3.7 PCB implementation-

We'll need to attach all the sensors to general-purpose PCB because we are ultimately connecting the sensors to hand gloves; that's why we need to build a compact system.

3.7.1 Collecting components

- Flex sensors 5
- Buttons 4
- JST pins 9
- General-purpose PCB
- Soldering station
- 10K resistors
- Female Pin headers
- Connecting wires

3.7.2 Circuit Connection-

By using FRITZING software-

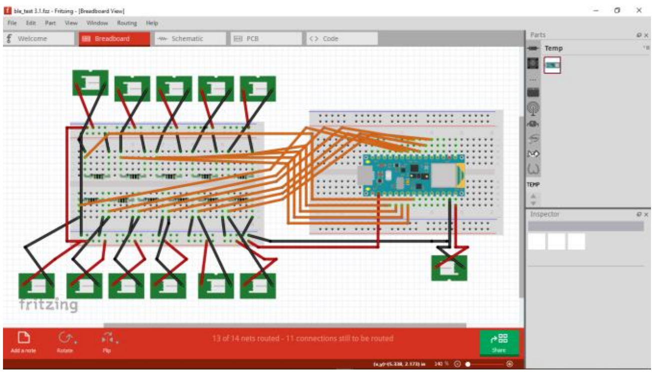

Figure 3.8 PCB schematic in breadboard

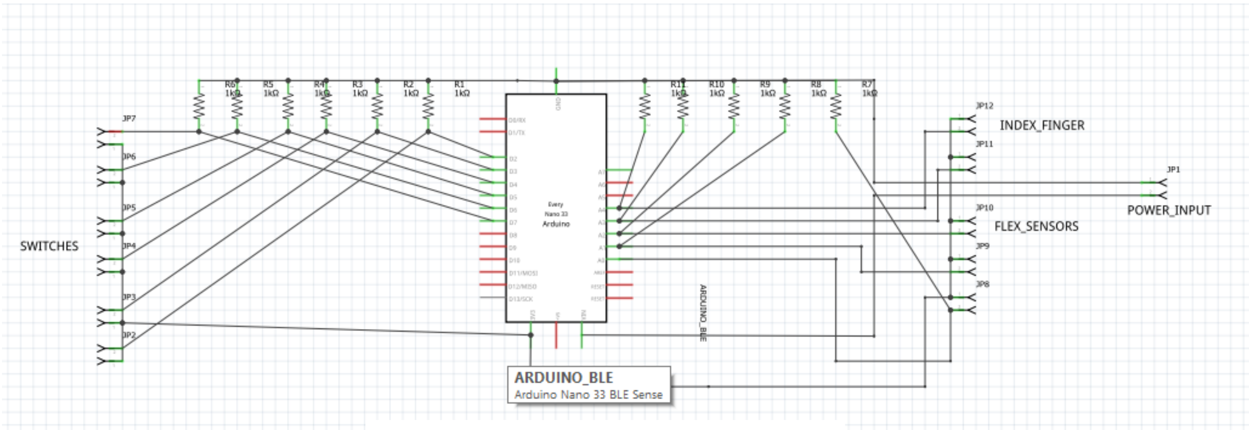

Figure 3.9 PCB schematic

3.7.3 Soldering-

After collecting all the required components, we need to solder them with PCB.

Figure 3.10 a. collecting tools

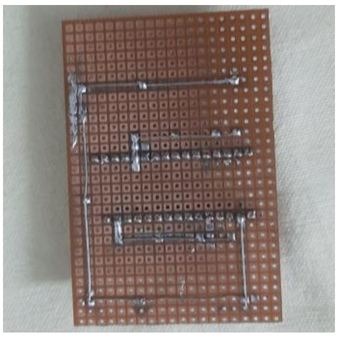

Figure 3.10. b.Bus Formation

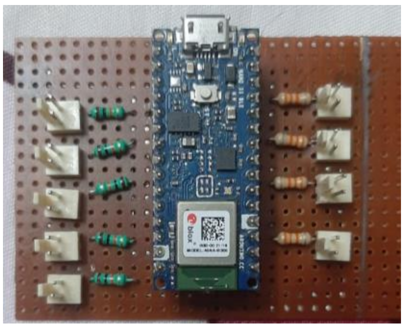

Figure 3.10.c. completed PCB top view

Figure 3.10.d. completed PCB bottom view

3.7.4 Attaching with Gloves

After the soldering process has been done, we need to attach all the sensors and PCB in Hand gloves like the 3d model we've created in chapter 3. So that we can wear it easily, refer image below.

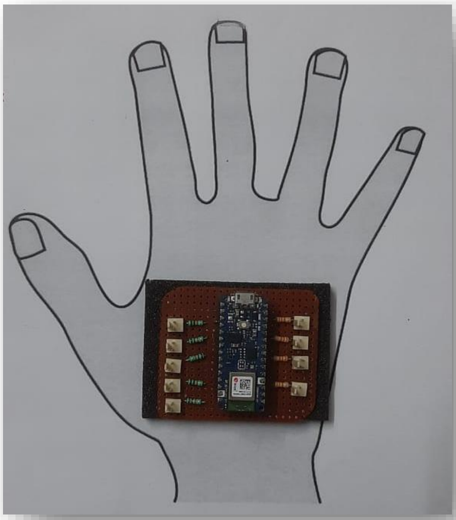

Figure 3.11 a Attaching PCB

Figure 3.11.b Attaching Flex

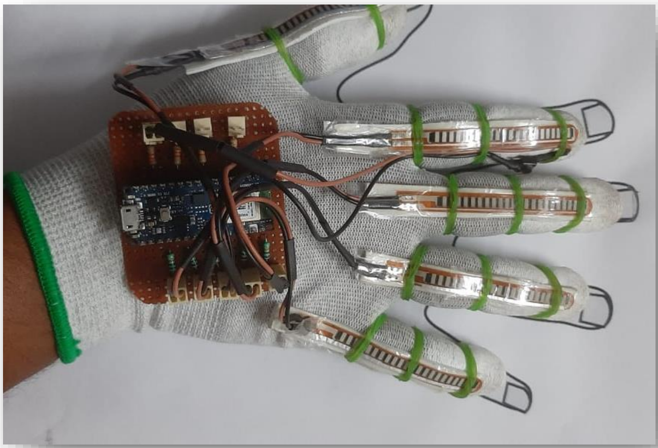

Figure 3.11.c Attaching whole pcb system

3.8 Working

After connecting all the components with the glove and microcontroller, we move to the coding part, where all the functionality of this system is given through Desktop. Also, by using an edge impulse Platform, we train our model so that the model gives the desired output, such as audio and video, with the highest accuracy. We can compare our model to demo outputs models.

3.8.1 Uploading Code-

Connect Arduino Nano BLE with a micro USB cable. Select COM port and click the upload button on Arduino IDE.

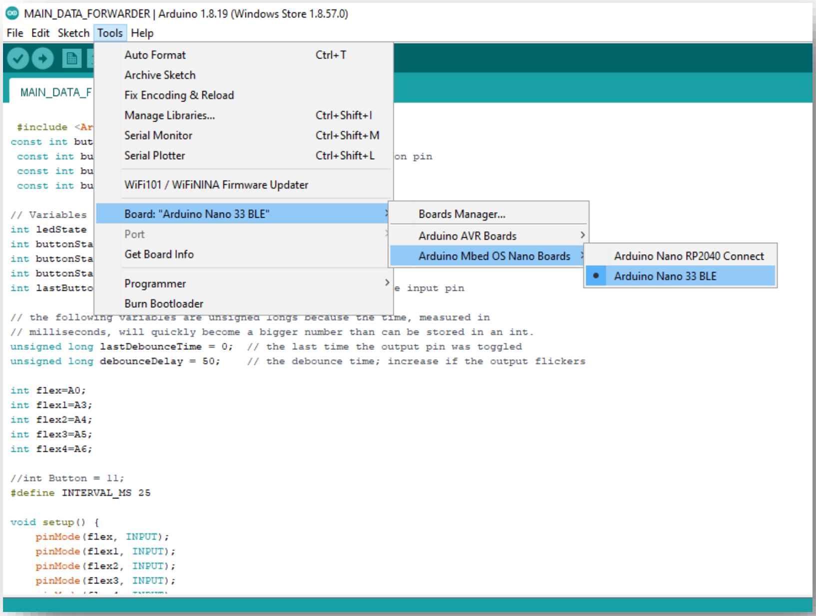

Figure 3.12 Uploading code

3.8.2 Machine Training-

To add a machine learning model to train our Arduino how to analyze sensor data.

And implement a training model for Arduino to achieve the highest accuracy. We need to connect our Arduino with sensors to the edge impulse platform.

3.8.3 Connecting to Edge impulse

(i) Create an edge impulse account

(ii) forward sensor data to edge impulse-

Using the command prompt, we connect our Arduino to the machine learning platform, as shown image below.

.3.8.4 Data Acquisition-

After connecting Arduino to edge impulse, the next step is to collect sensor data by clicking the start sampling button. as shown image below

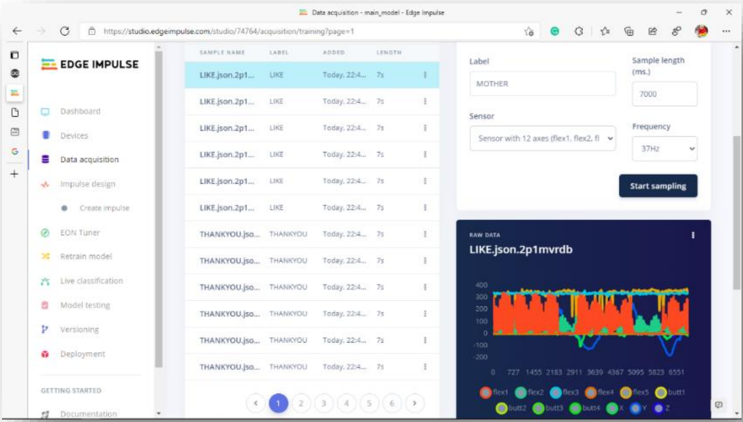

Figure 3.13 Data collection[20]

3.8.5 Adding learning Model

After collecting sufficient training data to edge impulse, we'll need to add a learning block to analyse training data. This can be done by using the impulse design Tab.

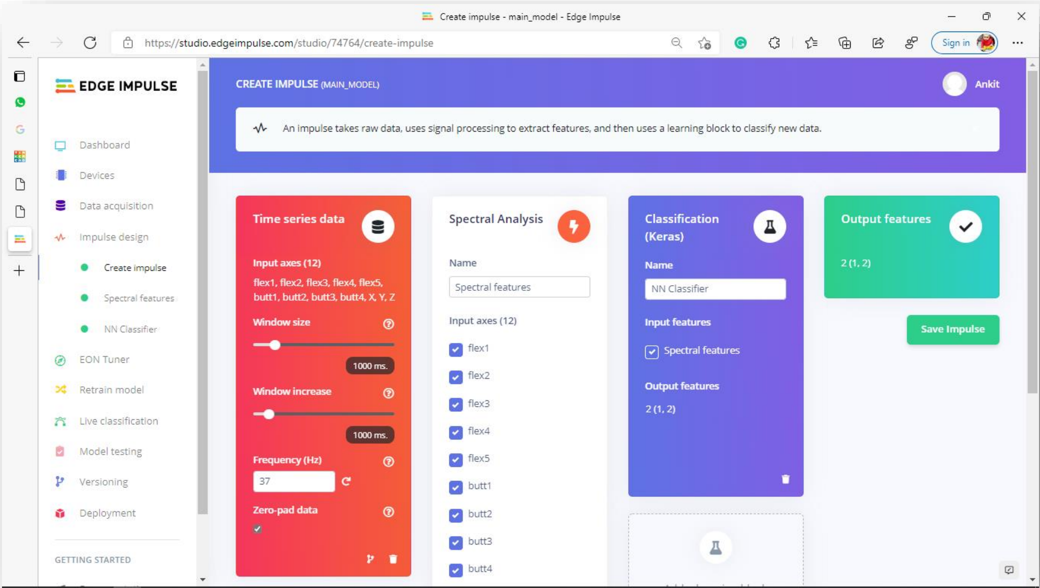

Figure 3.14 Adding learning model[20]

Link-[Dashboard - main_model - Edge Impulse]

3.8.6 Data classifier

After adding the learning model to our impulse, the next step is to generate features. This is done in generate feature Tab. After clicking generate feature button, we'll have a featured explorer where we can plot and demonstrate our data in 3D graphical form.

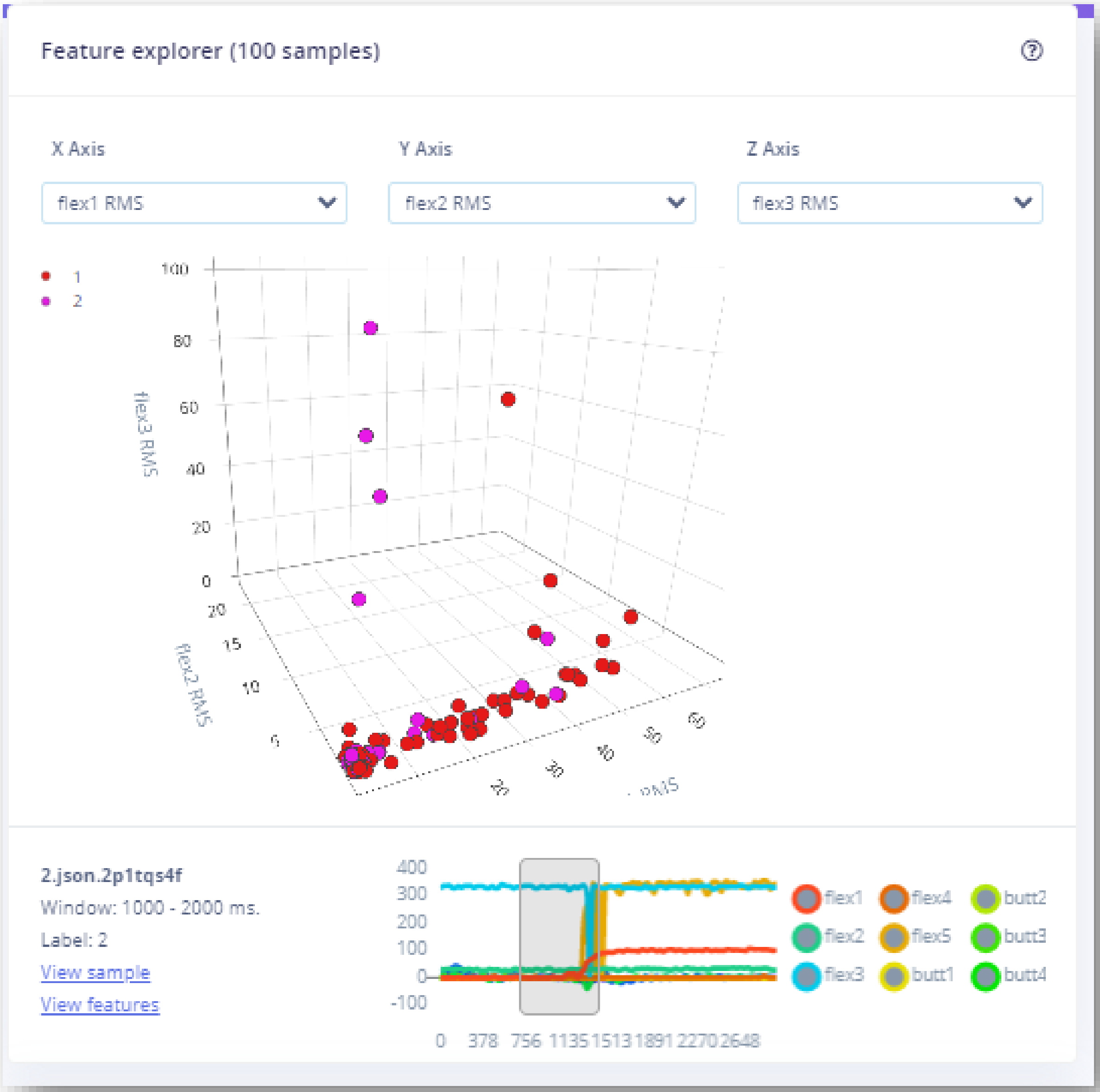

Figure. 3.15 graphical representation [20]

CHAPTER-4

RESULTS ANALYSIS AND DISCUSSIONS

4. Testing/Outputs

4.1 Accuracy testing-

In this chapter, we are going to explain how our model works with live data.

For this, you'll need to connect your Arduino to edge impulse, as discussed in chapter 5. After that, go to the live classification tab and start the sampling process, and it will collect sensor data from Arduino and analyse them with our crated training model. As you can see in this image.6.1.1 smart gloves perform a sign that actually is a "1" number, and similarly, image 6.1.2 shows smart gloves performing a number "2" sign.

After performing these signs, we move to accuracy analysis. Accuracy analysis shows us how accurate our training model is in predicting the desired outcome. We can start this by clicking the Star training button in the train model tab in edge impulse.

In image 4.1.3, we can see the accuracy performance of our training model, which shows our model has 80% of efficiency to predict the sign performed by smart gloves. This can be increased by adding more datasets to train our model.

figure.4.1 Performing '1' sign

figure. 4.2 performing '2' sign

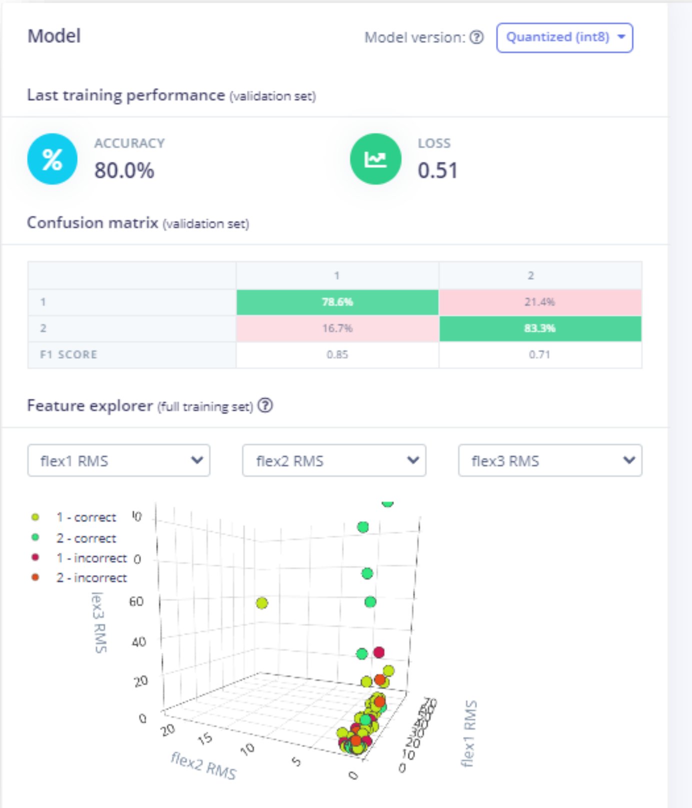

Figure 4.3 Accuracy testing[20]

4.2 Deploying to Arduino-

After successfully the machine learning part has been done with the highest accuracy, we need to deploy our machine learning model to our Arduino so that we can practically check the final outcome of this project.

Impulses can be deployed as an Arduino library. This packages all of your signal processing blocks, configuration, and learning blocks up into a single package. You can include this package in your own sketches to run the impulse locally. In this tutorial, you'll export an impulse and integrate the impulse in a sketch to classify sensor data.

Deploying your impulse

Head over to your Edge Impulse project, and go to **Deployment**. From here, you can create the full library, which contains the impulse and all external required libraries. Select **Arduino library** and click **Build** to create the library. Then download and extract the .zip File.
Then to add the library and open an example, open the Arduino IDE and:

1. **Choose** Sketch > Include Library > Add .ZIP library...
2. **Find the folder (do not go inside the folder), and select** Choose**.**
3. **Then, load an example by going to** File > Examples > Your project name - Edge Impulse > static buffer
4. Voila. You now have an example application that loads your impulse.

Figure. 4.4 deploying impulses to your Arduino

4.3 Results-

To see the output of the impulse, open the serial monitor from the Arduino IDE via Tools > Serial monitor and select baud rate 115,200.

This will run the signal processing pipeline and then classify the output.

Where we can further modify our Arduino code to employ according to our application need.

Follow the link to do so. -[On your Arduino (edgeimpulse.com)]()

Final outcome-

	1	2
1	78.6%	21.4%
2	16.7%	83.3%
F1 SCORE	0.85	0.71

figure. 4.5 Model Accuracy

4.4 Performing various sign-

Below image shows how we are performing different sign by using our smart gloves. All these signs are based on American sign language.

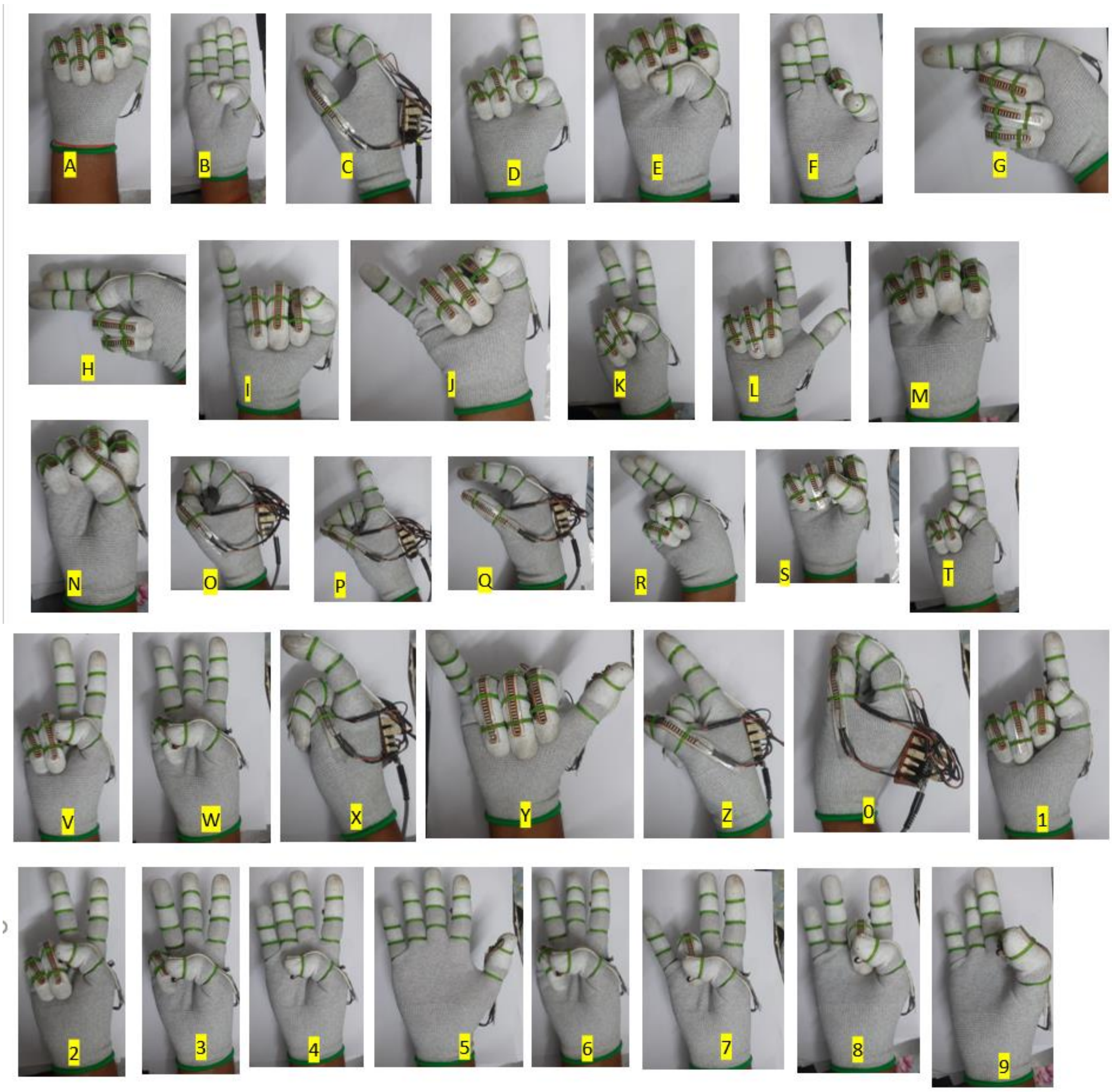

Figure.4.6 Sign language

4.5 OUTPUTs-

- In Relaxed Position-

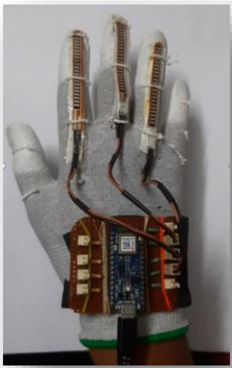

Figure 4.7 (a) Smart Glove

```
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
POSITION : RELAXED
```

Figure 4.7 (b) Serial Monitor

- Performing “1” sign-

Figure 4.8 (a) Smart Glove

```
POSOTION: 1
POSOTION: 1
POSOTION: 1
POSOTION: 1
POSOTION: 1
POSOTION: 1
POSOTION: 1
POSOTION: 1
```

Figure 4.8(b) Serial Monitor

- Performing “2” sign-

Figure 4.9 (a) Smart Glove

```
POSITION: 2
POSITION: 2
POSITION: 2
POSITION: 2
POSITION: 2
POSITION: 2
POSITION: 2
```

Figure 4.9(b) Serial Monitor

CHAPTER-5

CONCLUSIONS AND FUTURE SCOPE

In this chapter, we are discussing the final output of our project, future scope, application., Limitation of this project or area.

5.1 Final discussion-

In this project, we learned the following important points and technologies that are going to be beneficial for our future carrier:

- ✓ How to interference external sensors and electronic components to the microcontroller
- ✓ How to establish communication between microcontroller and computer.
- ✓ Arduino microcontroller
- ✓ Bluetooth LOW Energy
- ✓ How to code Arduino.
- ✓ How to simulate circuits
- ✓ How to use circuit simulation software tinker, cad
- ✓ How to design PCB layout
- ✓ How to use PCB design software fritzing
- ✓ How to use 3d model design platform tinker cad
- ✓ What is Machine learning
- ✓ How to collect data to train model
- ✓ How to train Machine
- ✓ How to implement machine learning model to Arduino
- ✓ How to deploy machine learning model to the microcontroller.
- ✓ How to use machine learning model for practical application.

5.2 Application:

Advantages & Applications

- Normal people don’t need to learn sign language to understand what the dumb people are trying to say.
- It is a portable device and can be easily carried anywhere
- The cost of the device is low.
- Power consumption for this system is also low.
- Easy interpretation
- Flexible to user
- This is a technology of the future that can be of great utility.
- In the future, this project will be useful for deaf and dumb people who cannot communicate with normal people. It is also useful for speech impaired and paralyzed patients means those who do not speak properly.

- This system eliminates the barrier in communication between the mute community and the normal people by discarding the need for an interpreter. It facilitates effective Real-time communication.
- It also provides communication between dumb and blind

5.3 Future scope-

- The glove output is limited to a single hand, and since hands differ in size and freedom of motion, therefore work can be done on this glove so that the wearable hand glove mouse can be flexible for every hand size.
- These gloves cannot capture facial expressions. So the device can be developed that can capture facial expression
- We could use key point matching and other techniques for more accurate decisions. The model can be run further for more epochs. The data could be increased. We can add words numbers as well for this. We could also do this by inputting a constant stream of images, and that get a resultant string for a particular word. We could also put an input field for letters and get the sign as an output, if possible, to make it into a fully sign communication platform. We could get an android application for the same. Text to speech could be added as well. It could be made multilingual as well

5.4 Limitation-

Although given the good performance of the metrics, there are a lot of limitations that exist in the solution. Firstly, the dataset is large, but even then, for computer-vision projects, it is relatively small; more data could be helpful. This is a difficult limitation to get around as ISL is varied and is two-handled. Further, the solution was limited to only the character and number subset in the ISL. Following this, there are various subsets of the ISL that are ambiguous and require context to clarify their meaning. Additionally, the model does not capture temporal inflections in ISL and does not perform any hand detection; this must be done by the user or by another solution added to the pipeline. Additionally, other methods such as Linear Discriminant Analysis (LDA) or even Convolutional Neural Networks (CNN) may prove to be useful.

- Accuracy and processing of the system may be slow. The data captured by the sensors may not be accurate. Even a small error in capture may lead to bizarre consequences.

- We will be developing only one hand's glove. More notably, users with differently sized hands/fingers, unable to fully control their fingers may, or unable to comfortably turn their wrists in all directions may have some difficulty in operating the glove. The flex sensor may not be at the correct position because of the difference in hand size.
- These gloves cannot capture the facial

5.5 Conclusion

Sign language is a useful tool to ease the communication between the deaf or mute community and normal people. Yet there is a communication barrier between these communities with normal people.

This project aims to lower the communication gap between the deaf or mute community and the normal world.

This project was meant to be a prototype to check the feasibility of recognizing sign language using sensor gloves.

With this project, deaf or mute people can use the gloves to perform sign language, and it will be converted into speech so that normal people can easily understand.

The main feature of this project is that the gesture recognizer is a standalone system, which is applicable in daily life

With this technology, Hand Gestures can be recognized with the CNN algorithm will provide us the best result other than using the sensors. Hand Gesture Recognition will provide two-way communication, which helps to interact between the impaired people to normal people without any difficulties.

REFERENCES AND APPENDICES

1. Nicholas Born.," Senior Project Sign Language Glove," ELECTRICAL ENGINEERING DEPARTMENT. California Polytechnic State University, 1-49,2010
2. Kirsten Ellis and Jan Carlo Barca. "Exploring Sensor Gloves for Teaching Children Sign Language. Advances in Human-Computer Interaction".1 8.,2012
3. Andreas Domingo, Rini Ameliorate, Kuang Ye Chow Pattern Matching for Automatic Sign Language Translation System using LabVIEW', International Conference on Intelligent and Advanced Systems 2007.
4. S2V(Hand Gesture Recognition and Voice Conversion System for Differentially Able Dumb People) SHOAIB AHMED V. Department of Electronics and Communication, Abdul Hakeem College of Engineering and Technology, Melvisharam, Vellore, Tamil Nadu 632 509, Submitted to Tech Expo-The Global Summit, London 2012.
5. Steven Douglas Collins. May," ADVERBIAL MORPHEMES IN TACTILE AMERICAN SIGN LANGUAGE. GRADUATE COLLEGE OF UNION INSTITUTE AND UNIVERSITY." 1131,2004.
6. Sign language Recognition System using Machine Learning Atreya Bain1, Shivam Birajdar2, Prof. Manonmani S.3 1,2Undergraduate Student, R. V. College of Engineering, Bangalore, Karnataka, India 3Assistant Professor, R. V. College of Engineering, Bangalore, Karnataka, India
7. A Survey on Hand Gesture Recognition Using Machine Learning N. Gopinath1, J. ANUJA2, S. ANUSHA3, V. MONISHA4
8. SIGN LANGUAGE DETECTION Pavitra Kadiyala1 1SCOPE, Vellore Institute of Technology, Vellore, India
9. ANDROID BASED PORTABLE HAND SIGN RECOGNITION SYSTEM Ponmandhira Muthu I1, Mohanapriya S2, Revathi M3 1Student, Computer Science and Science, Agni College of Technology, Chennai, India
10. Tinkercad | Create 3D digital designs with online CAD
11. https://docs.edgeimpulse.com/docs
12. https://en.wikipedia.org/wiki/Sign_language_glove
13. https://en.wikipedia.org/wiki/Machine_learning
14. https://en.wikipedia.org/wiki/Arduino
15. https://docs.arduino.cc/hardware/nano-33-ble
16. Sign Language Recognition Based on Intelligent Glove Using Machine Learning
17. https://www.researchgate.net/figure/The-26-letters-and-10-digits-of-American-Sign-Language-ASL_fig1_328396430
18. https://leonardocavagnis.medium.com/tinyml-machine-learning-for-embedded-system-part-i-92a34529e899
19. Flex Sensor: Pin Configuration, Working, Types & Its Applications (elprocus.com)
20. The Arduino Nano 33 BLE and BLE Sense are officially available! | Arduino Blog
21. Learn how to use Tinkercad | Tinkercad

Appendices

Code-Formation

```
/////////////////////////////////////////BASIC-CIRCUIT-CONNECTION-CODE//////////////////////////////////
// C++ code
//flex sensor

 int flex1=A1;
 int flex2=A2;
 int flex3=A3;
 int flex4=A4;
int flex5=A5;
const int threshold = 33;
int button_1=5;
int button_2=6;
int button_3=7;
int button_4=8;
int button_5=9;
int button_6=10;
void setup()
{
  Serial.begin(9600);

  pinMode(flex1, INPUT);
  pinMode(flex2, INPUT);
  pinMode(flex3, INPUT);
  pinMode(flex4, INPUT);
  pinMode(flex5, INPUT);
  pinMode(button_1, INPUT);
```

```
  pinMode(button_2, INPUT);
  pinMode(button_3, INPUT);
  pinMode(button_4, INPUT);
  pinMode(button_5, INPUT);
  pinMode(button_6, INPUT);

}
void loop()
{
  int value1=analogRead(flex1);//Thumb flex sensor
  int value2=analogRead(flex2);//index finger flex sensor
  int value3=analogRead(flex3);//middle finger flex sensor
  int value4=analogRead(flex4);//ring finger flex sensor
  int value5=analogRead(flex5);//little finger flex senors
  int value6=digitalRead(button_1); //Thumb Switch
  int value7=digitalRead(button_2); //Index finger switch
  int value8=digitalRead(button_3);//Contact switch
  int value9=digitalRead(button_4); // Ring finger switch
  int value10=digitalRead(button_5);// Middle contact switch
  int value11=digitalRead(button_6);//little finger switch

  //main function//
 if ((value1!=threshold) && (value2=threshold) && (value3=threshold) && (value4=threshold)
&& (value5=threshold) && (value6 == 1))
  {
   Serial.println("B");
    delay(1000);
  }
  else
```

```
    if ((value1=threshold) && (value2<threshold) && (value3<threshold) && (value4<threshold)
&& (value5<threshold))


        {
   Serial.println("A");
    delay(1000);
  }
  else
  Serial.println("RELAXED");
}
  ////////////////////////////////////////////END/////////////////////////////////////////////////////////////////////////
```

Code formation-

→For Flex sensors-

```
//////////////////////////////////////5FLEX-SENSORS////////////////////////////////////////////
int flex=A0;
int flex1=A3;
int flex2=A4;
int flex3=A5;
int flex4=A6;

void setup() {
   pinMode(flex, INPUT);
   pinMode(flex1, INPUT);
   pinMode(flex2, INPUT);
   pinMode(flex3, INPUT);
   pinMode(flex4, INPUT);

   Serial.begin(115200);
   Serial.println("Started");
}

void loop() {
  static unsigned long last_interval_ms = 0;
   int Flex_value;
   int Flex_value1;
   int Flex_value2;
   int Flex_value3;
   int Flex_value4;
   //float Button_value;
      Flex_value =analogRead(flex);
```

```
       Flex_value1=analogRead(flex1);
       Flex_value2=analogRead(flex2);
       Flex_value3=analogRead(flex3);
       Flex_value4=analogRead(flex4);
//
//        Serial.print(Flex_value);
//        Serial.print('\t');
//        Serial.print(Flex_value1);
//         Serial.print('\t');
//          Serial.print(Flex_value2);
//         Serial.print('\t');
//          Serial.print(Flex_value3);
//         Serial.print('\t');
        Serial.println(Flex_value4);
       Serial.print('\t');

    }
/////////////////////////////////////////////////////////END////////////////////////////////////////////////////////////////////////
```

- For Contact sensors (pushbutton)

```
//////////////////////////////////////////////4PUSHBUTTONS/////////////////////////////////////////////////////
const int buttonPin = 12;
const int buttonPin1 = 11;    // the number of the pushbutton pin
 const int buttonPin2 = 10;
 const int buttonPin3 = 9;

        // the current state of the output pin
int buttonState;             // the current reading from the input pin
```

```
int buttonState1;
int buttonState2;
int buttonState3;
int lastButtonState = LOW;   // the previous reading from the input pin

// the following variables are unsigned longs because the time, measured in
// milliseconds, will quickly become a bigger number than can be stored in an int.
unsigned long lastDebounceTime = 0;  // the last time the output pin was toggled
unsigned long debounceDelay = 50;    // the debounce time; increase if the output flicker
#define INTERVAL_MS 25
void setup() {
    pinMode(buttonPin, INPUT);
    pinMode(buttonPin1, INPUT);
    pinMode(buttonPin2,INPUT);
    pinMode(buttonPin3, INPUT);

    Serial.begin(115200);
    while (!Serial);
  Serial.println("Started");
}
void loop() {
  static unsigned long last_interval_ms = 0;



    if (millis() > last_interval_ms + INTERVAL_MS) {

        last_interval_ms = millis();

  int reading = digitalRead(buttonPin);
```

```
int reading1 = digitalRead(buttonPin1);
int reading2 = digitalRead(buttonPin2);
int reading3 = digitalRead(buttonPin3);

// check to see if you just pressed the button
// (i.e. the input went from LOW to HIGH), and you've waited long enough
// since the last press to ignore any noise:

// If the switch changed, due to noise or pressing:
if (reading != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
if (reading2 != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
if (reading3 != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
if (reading1 != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}

if ((millis() - lastDebounceTime) > debounceDelay) {
  // whatever the reading is at, it's been there for longer than the debounce
```

```
  // delay, so take it as the actual current state:

  // if the button state has changed:
  if (reading != buttonState) {
    buttonState = reading;
    if (buttonState == HIGH) {}
  }
  if (reading2 != buttonState) {
    buttonState = reading;
    if (buttonState == HIGH){}
  }
  if (reading3 != buttonState) {
    buttonState = reading;
    if (buttonState == HIGH) {}
  }
   if (reading1 != buttonState) {
    buttonState = reading;
    if (buttonState == HIGH) {}
  }
}

Serial.print(reading);
Serial.print('\t');
Serial.print(reading1);
Serial.print('\t');
Serial.print(reading2);
Serial.print('\t');
Serial.println(reading3);
Serial.print('\t')
```

```
}
}
/////////////////////////////////////////////////END///////////////////////////////////////////////////////////////////////////
```

→For Inbuilt Gyroscope Sensor

```
///////////////////////////////////////IN-BUILT-GYRO- ARUINO//////////////////////////////////////////
/*
  Arduino LSM9DS1 - Simple Gyroscope

  This example reads the gyroscope values from the LSM9DS1
  sensor and continuously prints them to the Serial Monitor
  or Serial Plotter.

  The circuit:
  - Arduino Nano 33 BLE Sense

  created 10 Jul 2019
  by Riccardo Rizzo

  This example code is in the public domain.
*/
#include <Arduino_LSM9DS1.h>

void setup() {
  Serial.begin(115200);
  while (!Serial);
  Serial.println("Started");
```

```
  if (!IMU.begin()) {
    Serial.println("Failed to initialize IMU!");
    while (1);
  }
  Serial.print("Gyroscope sample rate = ");
  Serial.print(IMU.gyroscopeSampleRate());
  Serial.println(" Hz");
  Serial.println();
  Serial.println("Gyroscope in degrees/second");
  Serial.println("X\tY\tZ");
}

void loop() {
  float x, y, z;

  if (IMU.gyroscopeAvailable()) {
    IMU.readGyroscope(x, y, z);

    Serial.print(x);
    Serial.print('\t');
    Serial.print(y);
    Serial.print('\t');
    Serial.println(z);
  }
}
/////////////////////////////////////////////////END//////////////////////////////////////////////////////////////////////////////
```

FINAL CODE—

```
////////////////////////////////ARDUINO-MAIN-DATA-FORWARDER///////////////////////////////////

 #include <Arduino_LSM9DS1.h>
const int buttonPin = 12;
 const int buttonPin1 = 11;    // the number of the pushbutton pin
 const int buttonPin2 = 10;
 const int buttonPin3 = 9;    // the number of the LED pin

// Variables will change:
int ledState = HIGH;         // the current state of the output pin
int buttonState;             // the current reading from the input pin
int buttonState2;
int buttonState3;
int lastButtonState = LOW;   // the previous reading from the input pin

// the following variables are unsigned longs because the time, measured in
// milliseconds will quickly become a bigger number than can be stored in an int.
unsigned long lastDebounceTime = 0;  // the last time the output pin was toggled
unsigned long debounce delay = 50;    // the debounce time; increase if the output flickers

int flex=A0;
int flex1=A3;
int flex2=A4;
int flex3=A5;
int flex4=A6;

//int Button = 11;
```

```
#define INTERVAL_MS 25

void setup() {
   pinMode(flex, INPUT);
   pinMode(flex1, INPUT);
   pinMode(flex2, INPUT);
   pinMode(flex3, INPUT);
   pinMode(flex4, INPUT);
   pinMode(buttonPin, INPUT);
    pinMode(buttonPin1, INPUT);
   pinMode(buttonPin2,INPUT);
   pinMode(buttonPin3, INPUT);

   Serial.begin(115200);
   while (!Serial);
  Serial.println("Started");

  if (!IMU.begin()) {
    Serial.println("Failed to initialize IMU!");
    while (1);
  }
}

void loop() {
  static unsigned long last_interval_ms = 0;
    float Flex_value;
    float Flex_value1;
    float Flex_value2;
    float Flex_value3;
```

```
 float Flex_value4;
  float x, y, z;
 //float Button_value;



 if (millis() > last_interval_ms + INTERVAL_MS) {



if (IMU.gyroscopeAvailable()) {
 IMU.readGyroscope(x, y, z);

     last_interval_ms = millis();

    Flex_value =analogRead(flex);
    Flex_value1=analogRead(flex1);
    Flex_value2=analogRead(flex2);
    Flex_value3=analogRead(flex3);
    Flex_value4=analogRead(flex4);

     Serial.print(Flex_value);
     Serial.print('\t');
     Serial.print(Flex_value1);
      Serial.print('\t');
       Serial.print(Flex_value2);
      Serial.print('\t');
       Serial.print(Flex_value3);
      Serial.print('\t');
       Serial.print(Flex_value4);
```

```
      Serial.print('\t');

int reading = digitalRead(buttonPin);
 int reading1 = digitalRead(buttonPin1);
int reading2 = digitalRead(buttonPin2);
int reading3 = digitalRead(buttonPin3);

// check to see if you just pressed the button
// (i.e., the input went from LOW to HIGH), and you've waited long enough
// since the last press to ignore any noise:

// If the switch changed, due to noise or pressing:
if (reading != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
 if (reading1 != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
if (reading2 != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
if (reading3 != lastButtonState) {
  // reset the debouncing timer
  lastDebounceTime = millis();
}
```

```
if ((millis() - lastDebounceTime) > debounceDelay) {
  // whatever the reading is at, it's been there for longer than the debounce
  // delay, so take it as the actual current state:

  // if the button state has changed:
  if (reading != buttonState) {
    buttonState = reading;

    // only toggle the LED if the new button state is HIGH
    if (buttonState == HIGH) {

    }
  }
  if (reading2 != buttonState) {
    buttonState = reading;

    // only toggle the LED if the new button state is HIGH
    if (buttonState == HIGH) {

    }
  }
  if (reading3 != buttonState) {
    buttonState = reading;

    // only toggle the LED if the new button state is HIGH
    if (buttonState == HIGH) {
```

```
      }
    }
  }

  Serial.print(reading);
  Serial.print('\t');
  Serial.print(reading2);
  Serial.print('\t');
  Serial.print(reading1);
  Serial.print('\t');
  Serial.print(reading3);
  Serial.print('\t');


    Serial.print(x);
    Serial.print('\t');
    Serial.print(y);
    Serial.print('\t');
    Serial.println(z);
    Serial.print('\t');
  }
}}
////////////////////////////////////////////////END////////////////////////////////////////////////////////////////////
```