MACHINE LEARNING ENHANCED SMART HAND-WEARABLE FOR SPEECH IMPAIRED INDIVIDUALS

A Project Report

Submitted by

ABIRAMI A (CB.EN.U4ECE20201)
DHARUN KUMAR S (CB.EN.U4ECE20213)
MATHIARASUN R J (CB.EN.U4ECE20232)
MITHUN E (CB.EN.U4ECE20234)

in partial fulfillment of the requirements for the award of

the degree of

Bachelor of Technology

in

Electronics and Communication Engineering

under the supervision of

Mr. Harikumar M E

Department of Electronics and Communication Engineering,
Amrita School of Engineering,
Amrita Vishwa Vidyapeetham,
Coimbatore – 641112

June 2024

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT, 1956

Dept of Elec. & Communication Engineering,
Amrita School of Engineering,
Amrita Vishwa Vidyapeetham,
Coimbatore – 641112

Certificate

This is to certify that this project titled **"Machine Learning Enhanced Smart Hand-Wearable for Speech Impaired Individuals"** submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Electronics and Communication Engineering**, by Ms. **Abirami A**, Mr. **Dharun Kumar S**, Mr. **Mathiarasun R J**, Mr. **Mithun E**, is a bonafide record of work carried out by them, under my supervision and that it has not been submitted, to the best of my knowledge, in part or in full, for the award of any other degree or diploma.

Mr. Harikumar M E
(Advisor)
Department of Electronics and
Communication Engineering,
Amrita School of Engg.,
Coimbatore

Dr. Shanmugha Sundaram G A
(**Batch Coordinator**)
BTech 2020-2024

Dr. Madhu Mohan N
(Dept. Chair)

Date :

This project was evaluated by us on ----------------------------------.

Internal Examiner

External Examiner

<u>Declaration</u>

We do hereby declare that this project titled **"Machine Learning Enhanced Smart Hand-Wearable for Speech Impaired Individuals",** submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Electronics and Communication Engineering**, is a true record of work carried out by us, under the supervision of **Mr. Harikumar M E** and that all information contained herein, which do not arise directly from my work, have been properly acknowledged and cited, using acceptable international standards. Further, we declare that the contents of this thesis have not been submitted, in part or in full, for the award of any other degree or diploma.

Coimbatore – 641112

Date :

ABIRAMI A
DHARUN KUMAR S
MATHIARASUN R J
MITHUN E

We would like to dedicate this work to the speech-impaired community, whose needs and experiences have driven the purpose of this work.

Acknowledgements

We would like to express our sincere gratitude to Mr. Harikumar M.E, Assistant Professor (Sr. Gr), for his dedication in guiding us throughout this project. His insights and support were instrumental in helping us stay on track and achieve our goals. We would like to thank the Chairperson of the Department of Electronics and Communication Engineering, Dr. N. Madhu Mohan for providing us with the right opportunities within the department for carrying out our project work. We express our gratitude to the members of our review panel for their insightful comments and feedback, which enabled us to improve our work. We would like to express our gratitude to every member of the departmental staff for their invaluable assistance and collaboration.

Abstract

Sign language is a fundamental communication method for individuals with speech impairments. While this technique offers significant benefits, it also presents several challenges that can hinder effective communication. Sign language, although rich and expressive, face limitations in terms of universal accessibility and understanding. Each country or region often has its own distinct sign language, leading to barriers in cross-cultural communication. Moreover, a lack of widespread knowledge and proficiency in sign language among the general population can isolate sign language users, making it difficult for them to interact with those who do not understand their form of communication. The sensor based Smart Hand-Wearable appears as an effective communication system that can bridge these gaps and enhance the interaction between speech-impaired individuals and the wider community.

A new method of communication, Finger Spelling, has been developed which allows users to convey messages with finger movements. Flex sensors are employed to record the bending intensity of each finger and match it to the preprogrammed gesture combination in the microcontroller board. A hybrid model is incorporated featuring 3 distinct modes, encompassing commonly used English phrases, English alphabets, numbers, and special characters. Based on the finger combination, the corresponding phrase is produced as a visual and an auditory output through a special android application designed for this purpose. A sensor signal dataset based on finger spelling is prepared. The data is collected from 100 individuals of varying age and gender, to capture the variation in finger bending intensity. Machine learning algorithms are developed, trained, and tested using the dataset samples and implemented on the Smart Hand-Wearable to enhance its performance.

A comparative analysis of the Smart Hand-Wearable’s performance, with and without machine learning implementation will be performed. Factors such as accuracy, recognition rate, response time, adaptability, user friendliness will be used to determine the necessity of employing machine learning in such smart devices to enhance its robustness in diverse environmental conditions and compatibility with varying user needs and preferences.

Table of Contents

List of Abbreviations

ADC	Analog-to-Digital Conversion
ISM	Industrial, Scientific and Medical
IDE	Integrated Development Environment
MIT	Massachusetts Institute of Technology
USB	Universal Serial Bus
ML	Machine Learning
CSV	Comma Separated Values

List of Symbols

R	Resistance
s	Sensitivity
r	Radius of Curvature

List of Tables

List of Figures

1. Introduction

Sign language recognition systems has emerged as a prominent area of research in recent years [1]. These systems convert gestures used in sign language into equivalent words or sounds [2], facilitating communication with individuals who are hearing or speech impaired. A specialized version of sign language, known as fingerspelling, includes alphabets, numerals, and special characters [3], serving as another valuable communication tool. There are two primary methods for detecting sign language gestures: image-based and sensor-based approaches. The image-based method involves acquiring images [4] or videos of gestures, which are then converted into voice output via speakers [5] or displayed on a smartphone app [5]. While several methods have been developed to enhance the accuracy and performance of these systems [7], [8], [9], challenges such as light variation, color dependence, and background issues persist. Despite these challenges, image-based systems are more cost-effective for real-time applications compared to sensor-based systems due to lower processing requirements, though they may be less comfortable for users [10].

In contrast, sensor-based systems have seen diverse and recent advancements. Some studies focus on Surface electromyography for the recognition of sign language, which captures the electrical activity of muscle tissues [11], [12], [13]. Other research has explored flex and inertial sensor utilization in combination [14], [15], [16], or individually, such as inertial [17] or flex [18] sensors. Notably, some studies have developed sensors specifically for measuring finger flexion, using materials such as fibers coated with Reduced Graphene Oxide (RGO) [19] or Pyrolytic Graphite Sheets (PGS) [20]. A notable gesture recognition system introduced in [21] includes four flex sensors, audio speakers, a Raspberry Pi 3 B+ model, and a regulated power supply, where each sensor corresponds to a distinct message. Additionally, a 3.5 mm audio jack is utilized for creating voices.

A proposed model [22] employs an Inertial Measurement Unit for gesture detection and five flex sensors for each finger, specifically targeting Indian Sign Language. The IMU, attached to the forearm, tracks hand movements in three-dimensional space, while flex sensors attached to a glove monitor finger orientation. Sensor inputs are processed by a Raspberry Pi, with the system referencing a pre-existing database. Upon user gesture input, if a match is found in the database, it is converted into audible voice output via a text-to-speech engine; otherwise, the system iterates to the next input. Similar to vision-based approaches, sensor-based methods offer numerous opportunities to enhance system accuracy and performance [14]. However, incorporating additional sensors, although beneficial for performance, significantly increases production costs and physical space requirements.

To mitigate these limitations, a sensor-based approach is suggested wherein each flex sensor functions as a binary unit. While it is possible to use flex sensors on both hands [23], this would exponentially increase the number of potential combinations, making it economically impractical. Therefore, the proposed method employs flex sensors on a single hand. Equipping each of the five fingers with a flex sensor creates five binary units, or bits, allowing for a maximum of 32 unique combinations. Assigning each combination to a specific alphabet obviates the need for additional sensors such as tactile, accelerometer, gyroscope, or pressure sensors, thereby reducing production costs and improving the smart glove's usability. Additionally, the proposed system features three operational modes. Another project [24] introduced a versatile system comprising two distinct modes: the first mode uses a flex sensor to control wheelchair movement, while the second mode combines a flex sensor and a voice module to generate voice output.

A smart glove capable of converting hand gestures into text is also proposed [25], which doubles as an obstacle detection system for visually impaired individuals. The glove features built-in wireless connectivity to facilitate emergency communication among the elderly. Another similar project [26] was enhanced with various operational modes to accomplish diverse tasks. In the current study, Mode 1 is tailored for frequently used expressions such as "Good morning" and "Thank you"; Mode 2 is reserved for alphabets in English; and Mode 3 is specialized for special characters. The resistance variation in the flex sensor [27] corresponds to the degree of finger bend, producing different values depending on whether the finger is bent or straight. These values are processed by the Arduino UNO, which houses an ATmega328P microcontroller, and displayed using one of three methods: LCD output [28], interfacing with a computer for recording or display [29], or presentation within an application [30]. For the proposed project, an Android application developed using the MIT App Inventor is employed for the presentation and audio output of the associated text, facilitated through Bluetooth module HC05 connectivity.

In [29], a system utilizing one inertial sensor and five flex sensors was implemented to identify the American Sign Language's 26 letters. Two pressure sensors were included in order to identify patterns of overlap and abduction between the middle and index fingers. To determine the shape of the hand, the flex sensor results were divided into areas that indicated whether the fingers were fully bent, slightly bent, or not bent at all. The standard deviation of the angular measurements from the inertial sensor was computed to detect hand motion. Using a Support Vector Machine (SVM) classifier, 98.2% accuracy was attained. The training set also contained a neutral gesture pattern that denotes a resting condition. A gesture is considered invalid if it is identified with less than 50% certainty. Five of the top machine learning-based classifiers are chosen for the proposed project, and their performance is assessed. With a trade-off between accuracy and memory requirements, the best classifiers include the random forest, bagging classifier, decision tree, additional trees classifier, and k-nearest neighbors classifier.

1.1 Objectives

The project emphasizes the development of a cost-effective solution by prioritizing sensor-based technology over vision-based alternatives [31]. This aims to streamline complexity and reduce overall costs, making the Smart Hand Wearables accessible to a broader audience.

To develop a flexible and adaptable Smart Hand Wearable. The objective is to enable users to express 32 distinct combinations for gesture-based communication by integrating five flex sensors that are strategically placed on the fingers. This flexibility ensures a nuanced and personalized means of expression.

To incorporate machine learning to refine gesture recognition of the Smart Hand Wearable. By collecting data from a diverse group of 100 individuals, the system is trained to adapt and evolve based on variations in flexing capabilities. This ensures accurate and personalized communication, addressing potential mismatches in gesture recognition.

2. Smart Hand-Wearable

2.1 Overview

The development of the proposed smart hand wearable is a critical and intricate aspect of the project. The design process required a meticulous balance of several key factors: cost, comfort, and reliability. Various models like the Intelligent Glove which contains ten flex sensors, a speaker, an SD card, an Arduino nano board [32], the Instrumented Glove, which is equipped with two touch sensors, one inertial sensor, and five flex sensors [33] were referred as shown in Fig. 2.1 and 2.2.

Initially, a woolen glove was employed for the purpose of enabling customized finger spelling recognition, aimed at helping communication for speech and hearing - impaired individuals. However, this first choice presented substantial challenges, particularly concerning user comfort and practicality, thereby needing an evaluation of the design.

In response to these challenges, the material of the glove was transitioned from wool to a combination of neoprene and polyester fabric. This modification was pivotal in addressing the need for a wearable that was not only functionally effective but also comfortable for the user.

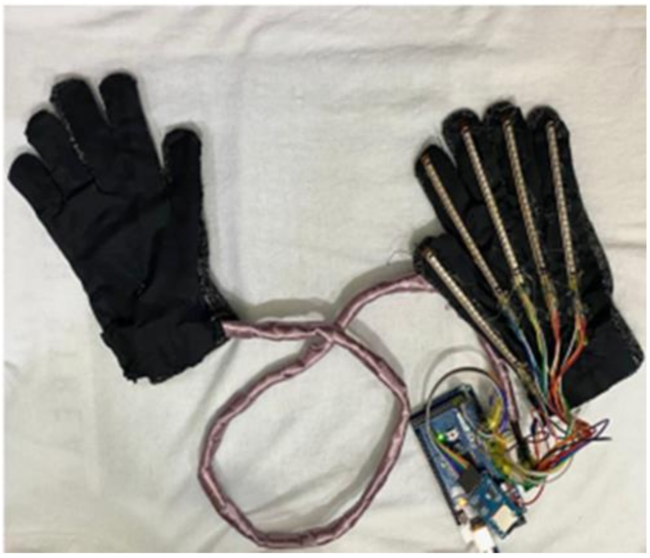


Fig. 2.1 Intelligent Glove prototype containing flex sensors, speaker, Arduino board

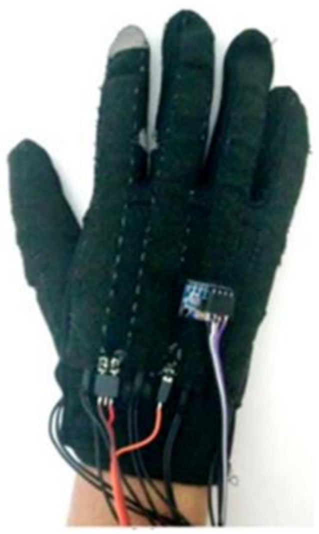


Fig. 2.2 Instrumented Glove's placement of flex, touch, and inertial sensors

2.2 Design

2.2.1 Sensor

In the exploration of sensor options for the envisioned model, guidance was sought from existing literature. One of the studies delves into various sensor types, evaluating their merits and demerits in the domain of wearable glove devices [34]. The study particularly underscores the efficacy of flex sensors, exemplified by a research glove named Action Sense, which exclusively employs this sensor type. The choice to incorporate flex sensors was primarily driven by considerations of cost-effectiveness. The following aspects were pivotal in advocating for the use of flex sensors in the proposed model.

Flex sensors provide a straightforward means of tracking finger movements, proving highly suitable for interpreting hand gestures owing to their precise measurement of finger bend. They show minimal power consumption compared to alternative sensor types, a critical factor in wearable devices where battery life is a concern. Being thin and flexible, seamlessly integrated into the glove without introducing bulk or discomfort. The resistance of flex sensors adjusts proportionally to finger bending, offering a precise method for capturing these movement.

In the context of the model, the design interprets the positions of five fingers (5 bits) for numeric, linguistic, and alphabetical representation. The absence of a requirement for wrist twisting or turning negates the need for added sensors like gyroscopes or accelerometers, streamlining the design and reducing system complexity.

2.2.2 Sensor Calibration

Flex sensor calibration is done to establish a reliable correlation between the sensor’s output and the bending intensity, ensuring accurate measurements. Over time, wear and tear on flex sensors can alter their electrical characteristics, making calibration crucial for accounting for such changes and preserving accuracy throughout the sensor’s lifespan. Calibration also helps in determining the sensitivity of the flex sensor.

Initially, the sensors were calibrated by bending it at different angles and recording the sensor output for the corresponding angle. Fig. 2.3 shows the initial calibration process. The issue with this process, as observed in the Fig. 2.3, was firstly that excess pressure was applied due to the fingers at the sensor ends which altered the sensor output, secondly bending of the sensor wasn’t uniform which lead to inaccurate output, and finally bending wasn’t correctly applied with respect to the marked angle value in the protractor.

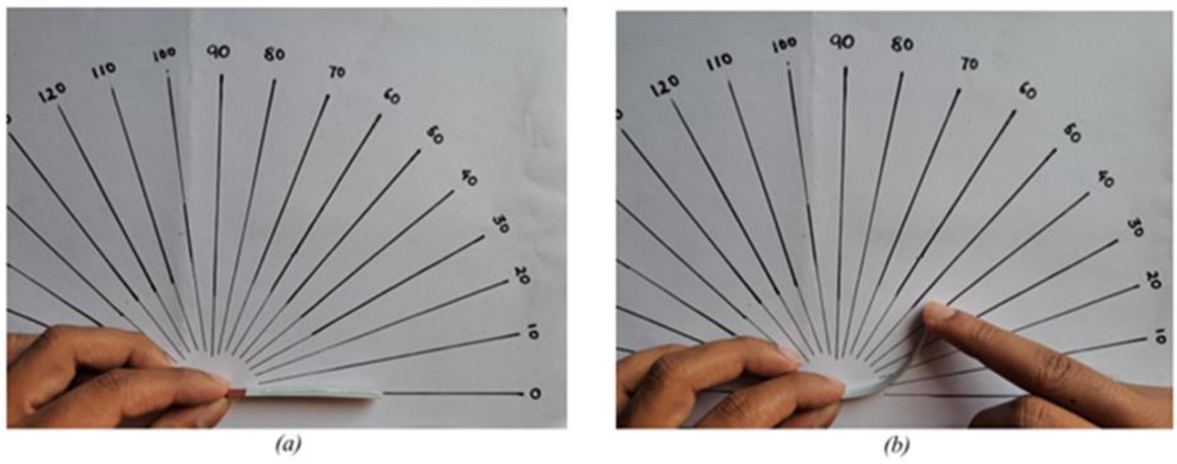


Fig. 2.3 Flex sensor bent at (a) 0 degree and (b) 40 degree

To obtain an accurate sensor calibration, a new setup was used which involved establishing a relation between the sensor output and the radius of curvature of the bent senor. For obtaining uniform bending, the sensors were made to bend across the edges of objects of uniform radius of curvatures. Fig. 2.4 shows the different objects, along with their radius of curvature, used for calibrating the sensor. A total of 11 objects were used of varying radius of curvature. An object of larger radius of curvature was approximately equivalent in modelling a smaller bending angle and an object of smaller radius of curvature was approximately equivalent in modelling a larger bending angle. Each sensor was individually calibrated by bending it across the edge of the object and its output was recorded.

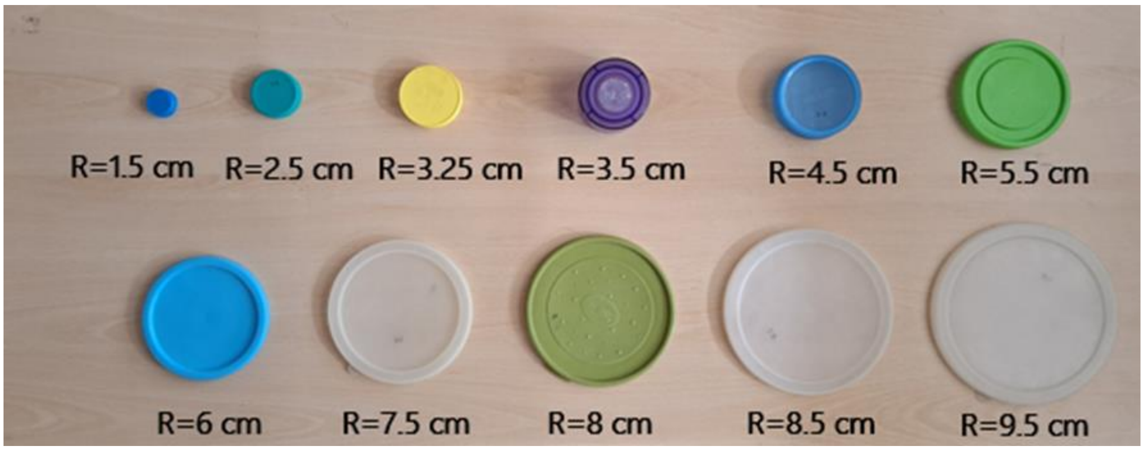


Fig. 2.4 Circular objects used for sensor calibration

2.2.3 Circuitry

A flex sensor functions as a tunable resistive element, exhibiting a directly proportional relationship between its electrical resistance and the degree of deflection it undergoes. To read the flex sensor output, it is combined with a resistor in series to form a voltage divider circuit [35] as shown in Fig. 2.5. The output is a variable voltage, obtained across the series resistor, which is read by the Analog-to-Digital Conversion (ADC) pins of the Arduino Uno microcontroller. The operational amplifier, LM324, acts as an impedance buffer in the circuit. Generally, flex sensors have relatively low impedance compared to the high input impedance of the ADC ports in the Arduino Uno microcontroller. This impedance mismatch can cause signal attenuation and noise susceptibility. The impedance buffer acts as an intermediary between the sensor and Arduino, resolving this mismatch [36]. This ensures accurate signal transmission from sensor to Arduino, and reduction in noise pickup. The voltage gain of the buffer is unity, therefore the signal obtained at the voltage divider circuit output is transmitted as such to the Arduino.

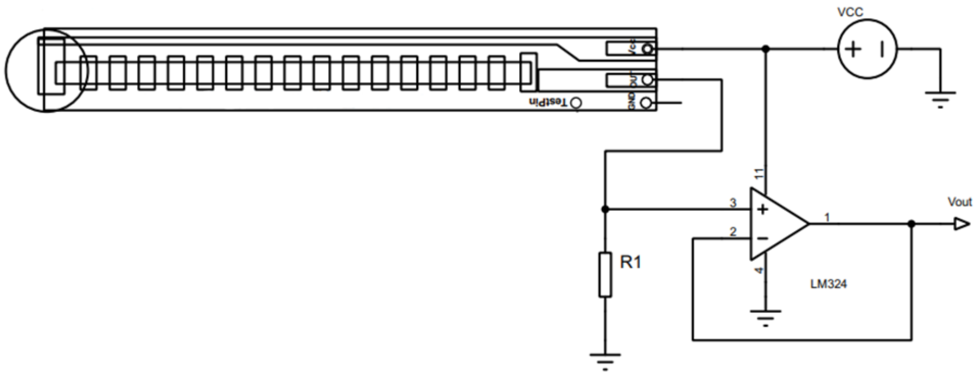


Fig. 2.5 Circuit schematic of Flex Sensor connection

Fig. 2.6 gives the complete circuit diagram with five flex sensor-opamp pairs respectively for five fingers of the human hand. The outputs from the 5 operational amplifiers are connected to the ADC ports of the Arduino Uno microcontroller. A separate Bluetooth module, HC-05, is used to transmit instructions from the Arduino to a Bluetooth compatible module for communication establishment with Smartphone.

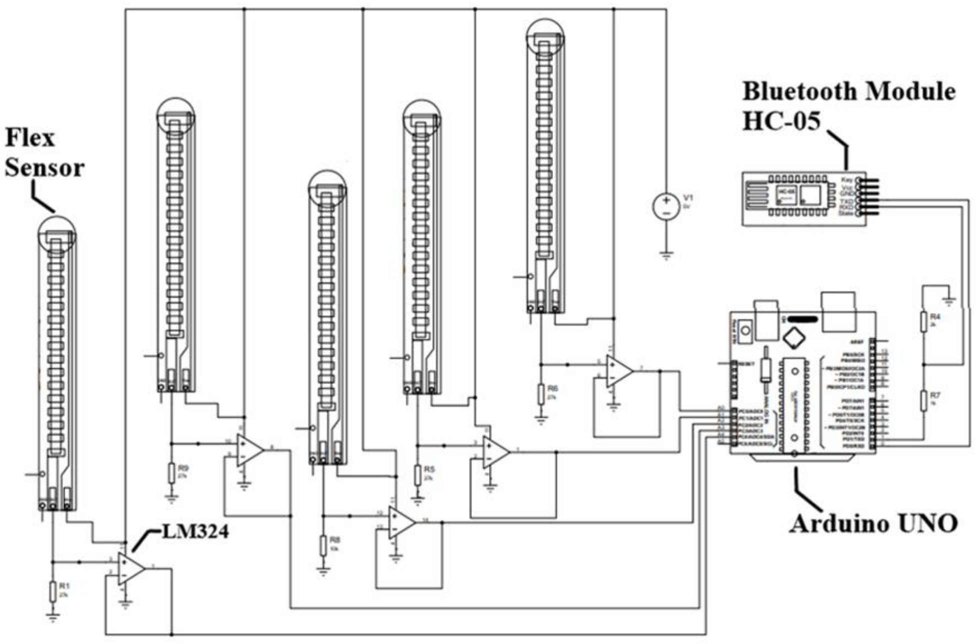


Fig. 2.6 Circuit schematic of the Smart Hand Wearable

2.3 Working

The glove's five fingers are equipped with five flex sensors, each of which is regarded as a bit that could contain a binary 0 or binary 1. When the sensor is in its rest state, its considered equivalent to binary 0 and when subjected to angular deflection, its considered equivalent to binary 1. Since there are 5 sensors which leads to five bits, 32 binary combinations are possible. Thumb finger acts as the Most Significant Bit and little finger acts as the Least Significant Bit. A particular combination is achieved by bending the associated fingers. The flex sensors detect the finger bending and the signals are sent to the microcontroller board. The corresponding binary combination is determined from the signals and the required output is sent to the Bluetooth module which transmits the signal to a Smart device.

A total of 3 modes are present in the Smart Hand Wearable, making it a hybrid model. Mode 1 consists of common English language phrases; Mode 2 consists of English language alphabets while Mode 3 consists of Numbers and Special Characters. Mode 1 is activated by the binary combination 01000 and Table 2-1 gives the binary combinations for the different phrases available in it. Mode 2 is activated by the binary combination 01100 and Table 2-2 gives the binary combinations for the alphabet choices available in it. Mode 3 is activated by the binary combination 01110 and Table 2-3 gives the binary combinations for the numbers and special characters available in it. 00000 indicates the process to stop and ask for whether to start or stop and 01111 indicates the process to start.

Table 2-1 Reference Table for Mode 1 Binary Combinations

Combination	**Output**
00000	Stop
00001	Good Morning
00010	Good Afternoon
00011	Good Evening
00100	Good Night
00101	See you then
00110	Thanks so much
00111	How was your day?
01000	MODE 1
01001	That’s interesting
01010	How's it going?
01011	How much does this cost?
01100	MODE 2
01101	I'm not sure
01110	MODE 3
01111	Start
10000	Can you help me?
10001	I am sorry
10010	Is there anything I can do?
10011	See you later
10100	I'm looking forward to
10101	I'm happy to hear that
10110	Can we reschedule?

10111	Let's keep in touch
11000	I need some feedback
11001	What’s the deadline?
11010	What's going on?
11011	How much does it cost
11100	I will be with you in a moment
11101	Please call me back
11110	My phone number is
11111	Have a nice day

Table 2-2 Reference Table for Mode 2 Binary Combinations

Combination	**Output**
00000	Stop
00001	A
00010	B
00011	C
00100	D
00101	E
00110	F
00111	G
01000	MODE 1
01001	H
01010	I
01011	J
01100	MODE 2
01101	K
01110	MODE 3
01111	Start
10000	L
10001	M
10010	N

10011	O
10100	P
10101	Q
10110	R
10111	S
11000	T
11001	U
11010	V
11011	W
11100	X
11101	Y
11110	Z
11111	

Table 2-3 Reference Table for Mode 3 Binary Combinations

Combination	**Output**
00000	Stop
00001	0
00010	1
00011	2
00100	3
00101	4
00110	5
00111	6
01000	MODE 1
01001	7
01010	8
01011	9
01100	MODE 2
01101	!
01110	MODE 3

01111	Start
10000	#
10001	$
10010	%
10011	^
10100	&
10101	*
10110	(
10111	)
11000	-
11001	
11010	+
11011	=
11100	?
11101	/
11110	>
11111	<

Initially all the fingers are straightened out, denoting the combination of 00000. If Mode 1 is to be activated, the index finger is bent while rest of the fingers are straightened out to achieve the combination of 01000. After activation, the user has to bend his/her fingers to obtained the required message. For example, to obtain the message ‘That sounds great’, the user has to bend thumb, middle, little finger and straighten out index, ring finger to achieve the combination of 10101. Similarly, to obtain the message ‘Can I help you’, the user has to bend thumb, index, ring finger and straighten out middle, little finger to achieve the combination of 11010. Mode 2 is activated by bending index, middle finger and straighten thumb, ring, little finger whereas Mode 3 is activated by bending index, middle, ring finger and straighten out thumb, little finger. Similar to Mode 1, the messages corresponding to Mode 2 or Mode 3 was obtained by bending the particular finger combinations.

3. Application Development for Smart Phone

3.1 Overview

The smart application enables the user to receive and display the text corresponding to the sign and has an in-built module to convert text-to-speech, thus providing both visual and audio information of the corresponding sign. The voltage values from flex sensors are converted to corresponding sign in text format and sent to the application via Bluetooth module in serial mode of communication and the text is shown in a text box and spoken out.

3.2 Bluetooth Module

One popular tool for wireless Arduino connection is the HC-05 Bluetooth module. Operating on Bluetooth 2.0 + EDR technology, it supports a range of about 10 meters. The module operates in the 2.4 GHz ISM (Industrial, Scientific, and Medical) frequency band. Its pinout includes Vcc, Gnd, Txd, Rxd, En, and State/Pin34, making it versatile for serial communication, and it's often used with the SoftwareSerial library present in Arduino Integrated Development Environment (IDE). The module operates at 3.3volts. The module's default communication baud rate is 9600 bits per second. The HC-05 is commonly employed in projects requiring wireless connectivity, offering a reliable and cost-effective solution.

3.3 Application Design and Working

The application works only in android versions of 2.1 and above. This Application is designed and built using Massachusetts Institute of Technology’s Application Inventor [37] as shown in Fig. 3.2. It consists of three modules, namely Bluetooth function, text box and text-to-speech converter. Fig. 3.1 shows the front-end display of the application.

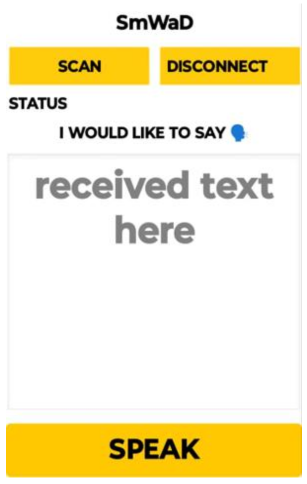
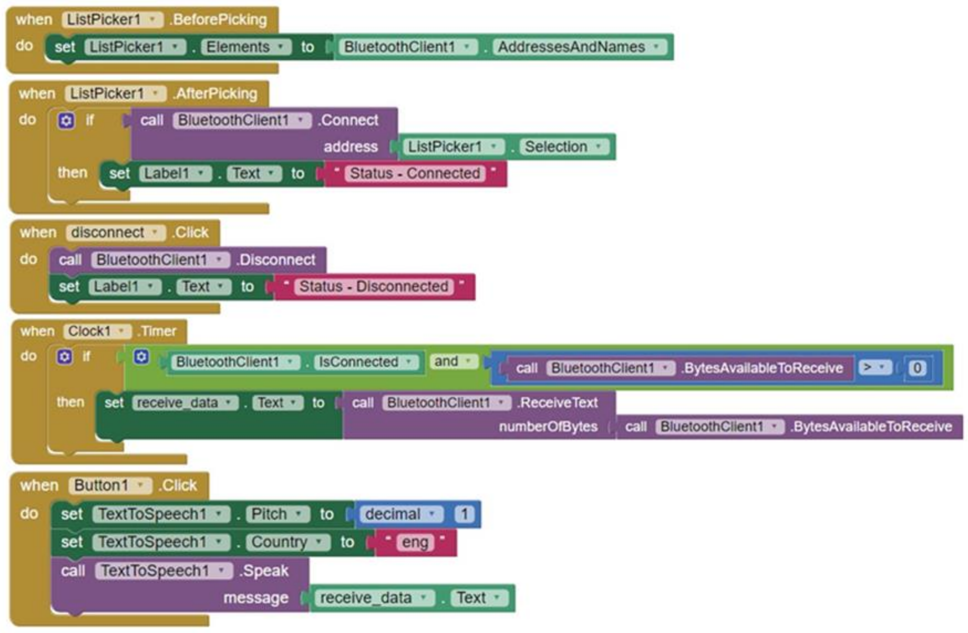


Fig. 3.1 Application Front-End Design



```
when ListPicker1 .BeforePicking
do  set ListPicker1 . Elements to BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do  if   call BluetoothClient1 .Connect
              address ListPicker1 . Selection
    then set Label1 . Text to " Status - Connected "

when disconnect .Click
do  call BluetoothClient1 .Disconnect
    set Label1 . Text to " Status - Disconnected "

when Clock1 .Timer
do  if   BluetoothClient1 . IsConnected and call BluetoothClient1 .BytesAvailableToReceive > 0
    then set receive_data . Text to call BluetoothClient1 .ReceiveText
                                         numberOfBytes call BluetoothClient1 .BytesAvailableToReceive

when Button1 .Click
do  set TextToSpeech1 . Pitch to decimal 1
    set TextToSpeech1 . Country to " eng "
    call TextToSpeech1 .Speak
         message receive_data . Text
```

Fig. 3.2 Application design block in MIT App Inventor

After connection with HC-05 Bluetooth module, the serial data from the Bluetooth function is read, stored in a variable called receive_data, which is used for displaying the text in text box and converting the text to speech in English language. The Application has only one screen to scan for available Bluetooth devices to connect, disconnect from the device, indicate the status of connection, display the text and speak the text displayed in textbox.

3.3.1 Bluetooth Function

The Bluetooth function consists of 2 buttons – "Scan" and "Disconnect" and a text box to display the status of the connection between the Bluetooth module and the Application. "STATUS- CONNECTED" is displayed when both the systems are connected and "STATUS- DISCONNECTED" is displayed when "Disconnect" button is pressed, disconnecting the systems.

3.3.2 Text Box

The text box continuously displays the serial data received using Bluetooth module. The variable received_data updates regularly on the received serial data. There is a clock facility by which the updation of received_data variable is controlled by providing one second delay.

3.3.3 Text-to-Speech Module

In-built module facility called text-to-speech module is employed to convert the text in text box to speech in English language with customized pitch. Whenever the “Speak” button is pressed, the module is invoked, and corresponding text is spoken out. Below show the blocks of code programmed through the visual programming interface of the Application, consisting of all the modules, synchronized together with the clock.

4. Dataset Collection

For training the different machine learning based models, a custom dataset is created by collecting finger flexing sensor data from a total of 100 volunteers.

4.1 Overview

For all the 32 combinations, ADC values from each flex sensor are collected using Arduino Uno and stored automatically in an excel sheet using python. Considering this as a multi-class problem, the values are arranged accordingly. Table 4-1 give the composition of data collection. The information from the Government of India's 2011 official census was used to determine the number of volunteers and the distribution of their ages.

Table 4-1 Composition of people for dataset collection

Age group (years)	Male	Female	Total
< 13	2	1	3
13-18	6	3	9
19-30	19	8	27
31-60	20	24	44
> 60	10	7	17
Total	57	43	100

4.2 Dataset Collection Process

The dataset collection process flow Application as shown in Fig. 4.1. Arduino IDE is used for collecting ADC values corresponding to the bending of each flex sensor using the hand wearable which is connected to a computer via Universal Serial Bus (USB) and the data is transmitted serially. Python script [38] is used for automatically storing data in csv format in an excel sheet for each person as a separate file. Name, age and gender are the information taken from each subject. Each subject receives a briefing regarding the data recording procedures prior to the recording process beginning. Before taking the readings for each sign, the subjects are instructed about it as they were not familiar with the indicators before to the study. Each time a data recording session starts, the individuals are asked to identify themselves by name, age, and gender. The laptop screen then prompts a five-second countdown to get ready. The individual has 20 seconds for each combination to prepare for the subsequent set of readings. Every gesture data instance is captured for a duration of one second, during which the subjects can effortlessly execute the motion once. In a single gesture recording session, this process is repeated 10 times.

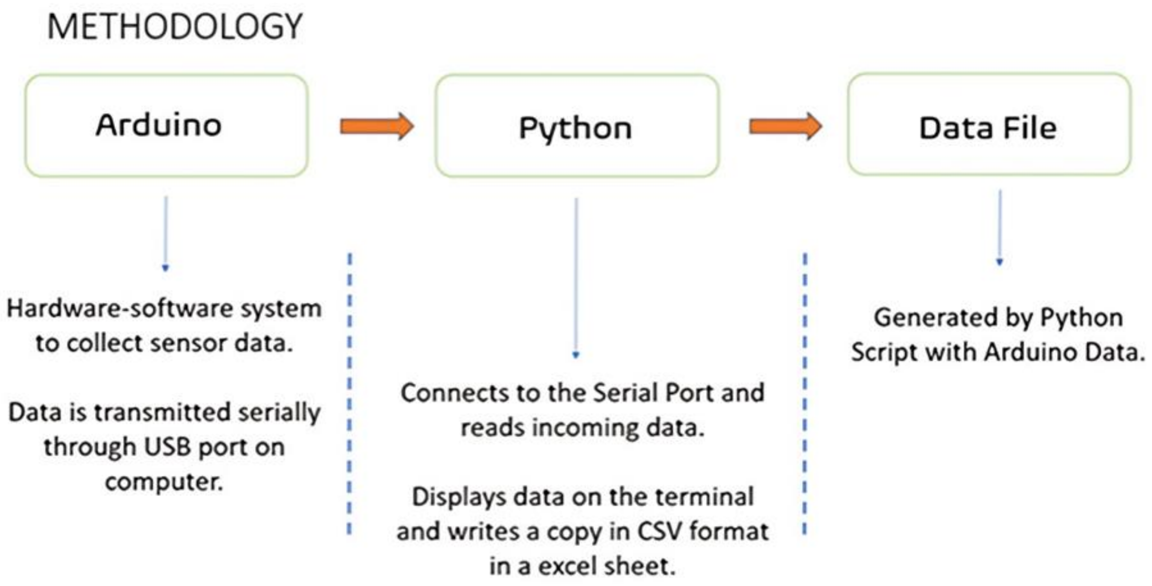

METHODOLOGY
Arduino
Python
Data File
Hardware-software system to collect sensor data.
Data is transmitted serially through USB port on computer.
Connects to the Serial Port and reads incoming data.
Displays data on the terminal and writes a copy in CSV format in a excel sheet.
Generated by Python Script with Arduino Data.

Fig. 4.1 Block diagram of the dataset collection process

There are 32 combinations of signs with 3 different modes, namely Mode 1 for phrases, Mode 2 for alphabets and Mode 3 for numbers and special characters. For a particular combination, 10 readings of ADC values from each flex sensor are taken from the sampler with the sampling frequency of 1Hz which sums up to 50 datapoints per combination. Therefore for 32 combinations performed by a single subject, 1600 datapoints are obtained. In total 1,60,000 datapoints are obtained from 100 subjects. Table 4-2 and 4-3 gives sensor readings of binary combination 00000 and 00001 collected from a subject.

Table 4-2 Sensor Readings for binary combination 00000

	Thumb finger	**Index finger**	**Middle finger**	**Ring finger**	**Little finger**
Test Case	**Binary Combination**				
	0	**0**	**0**	**0**	**0**
1	433	511	448	709	518
2	433	482	448	709	519
3	433	476	449	710	517
4	433	486	449	711	517
5	433	496	449	710	516
6	432	489	449	709	517

7	433	469	449	710	516
8	433	511	449	709	516
9	432	491	449	708	516
10	432	509	449	710	517

Table 4-3 Sensor Readings for binary combination 00001

	Thumb finger	**Index finger**	**Middle finger**	**Ring finger**	**Little finger**
Test Case	**Binary Combination**				
	0	**0**	**0**	**0**	**1**
1	431	517	463	709	503
2	431	517	464	709	503
3	431	517	463	708	503
4	431	516	463	709	504
5	431	517	463	709	504
6	431	517	463	709	504
7	431	516	463	710	503
8	431	516	463	709	503
9	431	517	463	709	503
10	431	517	463	709	504

After collecting from 100 people, considering it as a multi-class classification problem, a separate csv file is created which consists of 5000 datapoints arranged for each combination as a class, which is to be fed to machine learning algorithms for training, testing and validation.

The ADC value that is being recorded is calculated using the following formula,

$$ADC\ Value\ =\ \left(\frac{Analog\ Value}{Reference\ Voltage}\right) *\ Resolution \tag{4.1}$$

where the Analog Value is the reading from the flex sensor, Reference Voltage is the supply voltage of 5V provided to the flex sensor. The ADC port in Arduino Uno has a 10-bit resolution which is equal to 2^{10} levels i.e., 1024 levels. The ADC value is a zero-based index therefore its range is from 0 to 1023.

5. Dataset Preparation

The data preparation phase is crucial for ensuring that the dataset is appropriately formatted and preprocessed for subsequent analysis. By extracting relevant features, encoding the target variable, standardizing the data, and splitting it into training and testing sets, we establish a solid foundation for building and evaluating classification models effectively.

5.1 Feature Extraction

The dataset under consideration contains information related to finger positions, crucial for classification tasks. Initially, the dataset is loaded, and the features pertinent to finger positions are extracted. These features are fundamental for understanding and predicting various aspects of the data.

5.2 Target Variable Encoding

Accompanying the features is a target variable, which provides labels for the data samples. However, these labels are stored as strings, necessitating conversion into a numerical format for model compatibility. We help the model understand and learn from the given data by converting the string labels into their numerical equivalents.

5.3 Standardization

Standardization is used to make sure that characteristics are consistent and comparable. The characteristics are scaled in this manner to have a zero mean and a one standard deviation. By bringing all features to the same scale, this kind of standardization aids in both stabilizing and speeding up the learning process of machine learning algorithms.

5.4 Data Splitting

A training set and a testing set are created from the dataset to properly evaluate the classification model's performance. The model is trained using the training set, which consists of most of the data. In the meantime, the testing set—which is a smaller subset of the data—acts as a stand-alone dataset for assessing the effectiveness of the model. Thirty percent of the data was used for testing, while seventy percent was used for training.

6. Machine Learning Implementation

6.1 Need for Machine Learning

As individuals age, significant alterations take place in their neuromuscular system, which we have discussed earlier in dataset collection, leading to a decrease in hand dexterity, flexion, and strength. Consequently, the ADC (analog-to-digital converter) values vary from person to person as the voltage produced in device varies due to the flex they produce differs. To address this variability and cater to users across different age groups, we employ machine learning techniques. By doing so, we ensure that the output remains consistent for all gestures, regardless of the user's age category or the ADC values they generate.

6.2 Machine Learning in Biomedical Signaling

Machine learning (ML) is increasingly being used in biomedical signal processing to extract meaningful information from complex biological signals but in our work, we have used to extract output from the Voltage signal converted to digital produced by flexing of the fingers [39].

Classification: ML algorithms can classify biomedical signals into different categories, such as normal and abnormal patterns. In our work we have classified signal based on the output produced by different fingers by flexing combing it to make a pattern to give its equivalent output

Feature Extraction: ML techniques can automatically extract relevant features from biomedical signals, reducing the dimensionality of the data while preserving important information. These features can then be used for subsequent analysis or classification tasks.

Signal Denoising and Enhancement: ML models can effectively denoise biomedical signals by learning the underlying patterns of noise and separating it from the signal of interest. This improves the quality of the signals for accurate analysis.

Prediction and Forecasting: ML algorithms can analyze temporal patterns in biomedical signals to predict future outcomes.

Signal Reconstruction: ML techniques can reconstruct missing or corrupted parts of biomedical signals using available data, enabling the recovery of valuable information in scenarios where signal quality is compromised.

6.3 Machine Learning Algorithms Deployed

We explored following algorithms to find the best fit for your project, balancing accuracy and computational efficiency, as a prudent approach. We have assessed the performance of each algorithm using appropriate evaluation metrics relevant to our problem, such as accuracy, precision, recall, F1-score and Memory.

6.3.1 K-Nearest Neighbors (KNN)

A straightforward and user-friendly machine learning approach for classification and regression applications is K-Nearest Neighbors (KNN). Its lack of explicit model learning during the training stage puts it in the class of instance-based or lazy learning algorithms. Rather, it retains the complete training dataset in memory and generates predictions by comparing fresh data points to the preexisting training examples [40]. In our work we have employed the K-Means algorithm to identify distinct clusters within our dataset. After preparing the data and selecting the number of clusters (k=3), K-Means clustering was applied to group similar data points based on their features. The algorithm successfully partitioned the data into three clusters, each representing data points with similar characteristics as shown in Fig. 6.1. This process was repeated to each dataset of the five fingers. The important parameters are n of nearest neighbors set to 5, weights set to uniform, and algorithm set to auto.

6.3.2 Decision Trees

A machine learning approach called a decision tree is utilized for both regression and classification problems. To create homogenous subsets that are more predictive of the target variable, it works by recursively partitioning the data into subsets based on the values of input attributes. It is arranged in a tree-like fashion, with each internal node denoting a test or decision made on a particular characteristic or attribute, each branch denoting the decision's result, and each leaf node representing the choice or prediction made in the end [41]. The tree is built in a top-down, recursive manner. At the outset, all the training examples are placed at the root of the tree. The attributes used for partitioning are categorical. If the attributes are continuous-valued, they are converted into categorical ones before they are used. The examples are then partitioned recursively based on the chosen attributes which is represented in Fig. 7.2. The test features are chosen using a heuristic or statistical measure, like knowledge gain. The partitioning may cease under one of three circumstances. When all the samples for a specific node are in the same class, that is the first requirement. The second requirement is that no more qualities can be divided; in this scenario, the leaf is classified by majority vote. When there are no more samples available is the third need. The important parameters are criterion set to Gini impurity and random state set to 42.

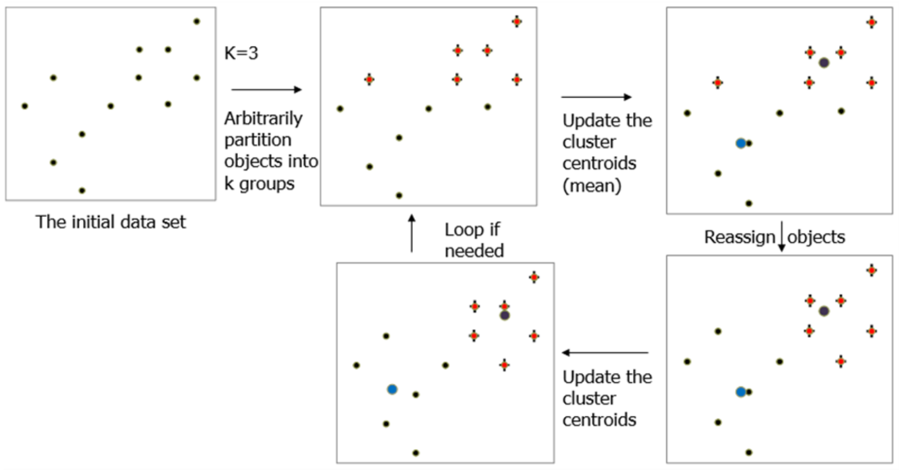

K=3
Arbitrarily partition objects into k groups
The initial data set
Update the cluster centroids (mean)
Reassign objects
Loop if needed
Update the cluster centroids

Fig. 6.1 Implementation of K-Mean algorithm

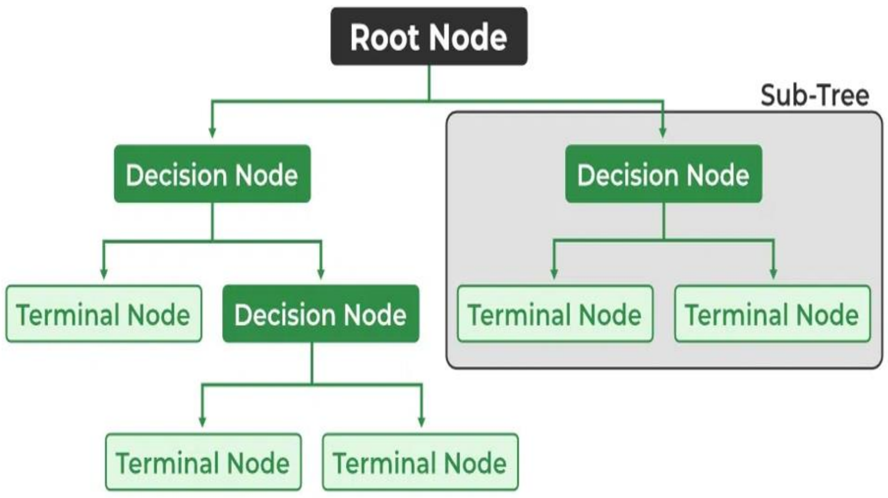


Fig. 6.2 Implementation of Decision Tree algorithm

6.3.3 Random Forest

A potent ensemble learning technique for both regression and classification applications is called Random Forest. During training, it builds many decision trees, and the final prediction is the mode (for classification) of each individual tree. A collection of cooperative decision trees is called a random forest. Each tree votes on a prediction, and the final answer is the most popular vote (for categories) or the average (for numbers) as shown in Fig. 7.3. This makes it more accurate and less prone to mistakes than a single tree.

In a Random Forest, bootstrapping involves sampling a random subset of the dataset with replacement to create multiple bootstrap samples, which are then used to train decision trees. By guaranteeing that only a random subset of characteristics is taken into account for splitting at each node, feature randomness lowers correlation between trees and increases variety. Each decision tree is grown to its maximum depth without pruning, resulting in a collection of deep and diverse trees. For classification tasks, the class predicted by most trees is chosen as the final prediction. Random Forest offers advantages such as robustness to overfitting by averaging predictions, providing feature importance measures, parallelization for efficient training on large datasets, and handling imbalanced data through class weights or sampling techniques [41]. The important parameters are criterion set to Gini impurity, max features set to sqrt, no of estimators set to 200, and random state set to 42.

6.3.4 Extra Trees Classifier

For classification tasks, the Extra Trees Classifier—also known as the Extremely Randomized Trees Classifier—is a sophisticated ensemble learning technique. This method aggregates multiple decision trees, each constructed from randomly selected subsets of the data and features. Unlike traditional decision trees, which aim to find the optimal splits based on information gain, the Extra Trees Classifier introduces additional randomness by choosing split points at random. This helps in reducing overfitting and improving the model's generalizability.

In the development of our Smart Hand Wearable, the Extra Trees Classifier is pivotal for refining gesture recognition. The classifier builds an ensemble of decision trees by using random subsets of each tree's characteristics and data points for training as depicted in Fig. 6.4. This diversity among the trees ensures that the model is robust and accurate when interpreting the sensor data from the wearable. The Extra Trees Classifier is particularly well-suited for our application due to its efficiency in handling high-dimensional data. The Smart Hand Wearable relies on data from five flex sensors strategically placed on the fingers, capturing detailed movements and positions. This data is complex and high-dimensional, and the Extra Trees Classifier can process it effectively, ensuring that subtle variations in gestures are recognized accurately.

Moreover, the Extra Trees Classifier's robustness to noise and variations in the dataset is critical for our project. We collected data from a diverse group of 100 individuals, each with unique flexing capabilities. The classifier’s ability to generalize well across different users ensures that the Smart Hand Wearable provides accurate and personalized communication, adapting to individual differences in gesture performance.

Another significant advantage of the Extra Trees Classifier is its speed in both training and prediction phases. This efficiency is beneficial for real-time applications like the Smart Hand Wearable, where quick response times are crucial. Fast training allows for the model to be updated efficiently as new data is collected, and rapid predictions ensure that users experience minimal delay between making a gesture and the system recognizing it, enhancing the overall user experience. The important parameters are criterion set to Gini impurity, maximum features set to sqrt, no of estimators set to 200, and random state set to 42.

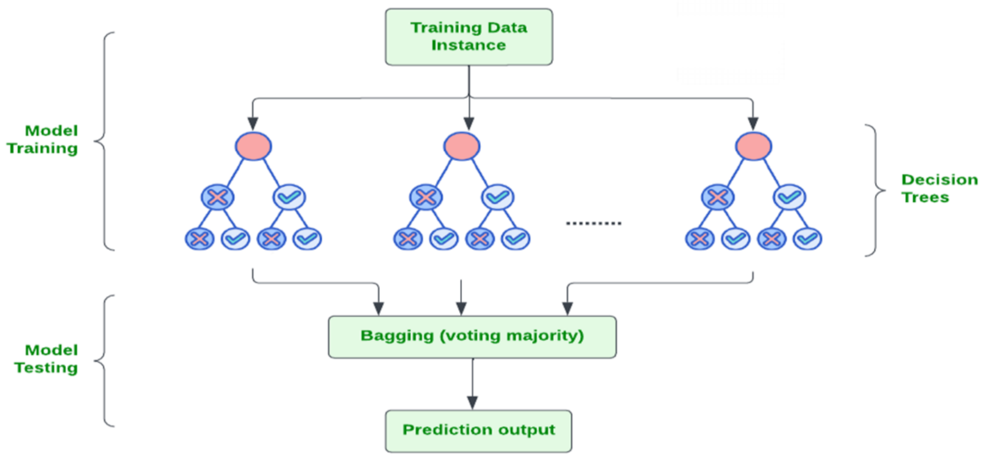


Fig. 6.3 Implementation of Random Forest algorithm

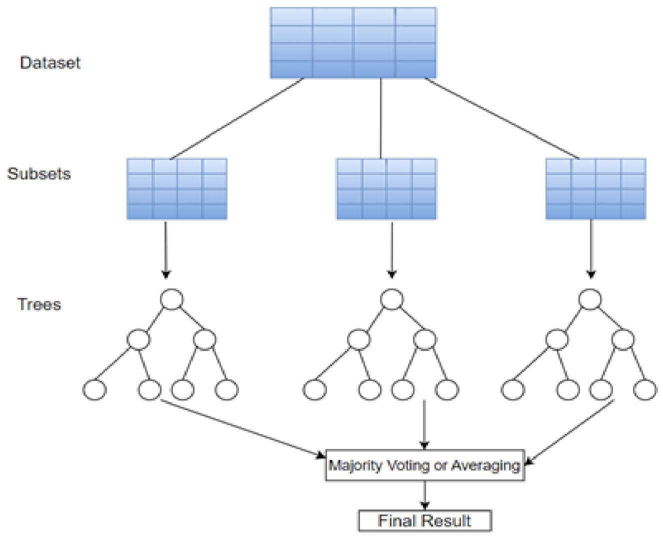


Fig. 6.4 Implementation of Extra Tree Classifier

6.3.5 Bagging Classifier

Bootstrap Aggregating, also known as the Bagging Classifier, is a potent ensemble learning technique that is intended to improve the accuracy and stability of machine learning models. It functions by creating several model iterations and integrating their forecasts. Bootstrapping is the process of creating a random subset of the original training data for each model to be trained on. This technique helps reduce variance and prevents over-fitting, making the Bagging Classifier a robust choice for various classification tasks. A collection of decision trees is produced by the classifier, and each tree is trained using a distinct bootstrapped subset of the data. By aggregating the predictions of these trees, the Bagging Classifier delivers a more accurate and reliable model compared to individual decision trees. The important parameters are base estimator set to Decision Tree Classifier, no of estimators set to 200, and random state set to 42.

Moreover, the Bagging Classifier's robustness to noise and variations in the dataset is critical for our project. We collected data from a diverse group of 100 individuals, each with unique flexing capabilities. The Bagging Classifier’s ensemble approach ensures that the model generalizes well across different users, providing accurate and personalized communication by adapting to individual differences in gesture performance.

Another significant advantage of the Bagging Classifier is its efficiency in both training and prediction phases. The parallel nature of the bagging process allows for training multiple models simultaneously, which speeds up the overall training time. This efficiency is particularly beneficial for real-time applications like the Smart Hand - Wearable, where quick response times are crucial. Fast predictions ensure that users experience minimal delay between making a gesture and the system recognizing it, enhancing the overall user experience. Additionally, the Bagging Classifier's ensemble method naturally mitigates the risk of over-fitting. By averaging the predictions of multiple trees, the classifier balances the biases and variances of individual models, resulting in a more stable and generalizable model. This characteristic is essential for our wearable, which needs to interpret gestures accurately in various real-world scenarios and environments.

7. Results and Discussion

Results of flex sensor calibration, working of the smart hand wearable and the dataset collected are discussed.

7.1 Flex Sensor Calibration

As mentioned in the report earlier, each of the 5 flex sensors were individually calibrated against a set of 11 circular objects of varying radius of curvature. Each calibration was repeated for 10 times. An average of the measured resistance values of the Thumb Finger Flex Sensor are shown in Table 7-1. For the same sensor, Fig. 7.1 gives the Resistance vs Radius of Curvature plot.

Table 7-1 Resistance Variation with Radius of Curvature for Thumb Finger Flex Sensor

Radius of Curvature (cm)	**Resistance (kΩ)**
1.50	35.34 ±0.01
2.50	34.86 ±0.01
3.25	34.53 ±0.02
3.50	34.41 ±0.03
4.50	34.01 ±0.02
5.50	33.64 ±0.03
6.00	33.47 ±0.01
7.50	33.09 ±0.03
8.00	32.91 ±0.02
8.50	32.78 ±0.02
9.50	32.55 ±0.01

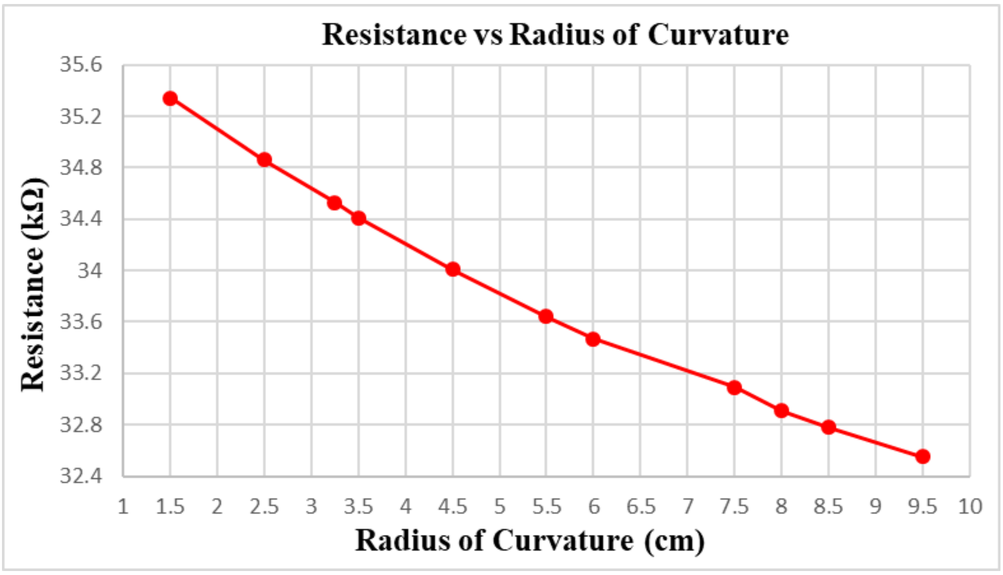


Fig. 7.1 Resistance vs Radius of Curvature plot for Thumb Finger Flex Sensor

The graph obtained in Fig. 7.1 is a linear plot of the form,

$$R = sr + c \tag{7.1}$$

where the variable R denotes Resistance of the Flex Sensor, r denotes Radius of Curvature, s denotes the slope and c denotes the y-axis intercept of the linear plot. As the radius of curvature increases, the resistance of the sensor shows an almost linear decrease. This implies as the radius of curvature increases, the bending angle of the sensor decreases and the resistance decreases. The sensitivity of the sensor is obtained from the slope of the graph, and in this case it’s approximately -0.35 kΩ/cm. The negative sign indicates that resistance of the sensor decreases by an amount of 0.35kΩ for every 1cm increase in the radius of curvature. Similarly, Table 7-2, 7-3, 7-4, 7-5 give the measured resistance values of the Index Finger Flex Sensor, Middle Finger Flex Sensor, Ring Finger Flex Sensor, Little Finger Flex Sensor respectively.

Table 7-2 Resistance Variation with Radius of Curvature for Index Finger Flex Sensor

Radius of Curvature (cm)	**Resistance (kΩ)**
1.50	29.1 ±0.02
2.50	28.97 ±0.01
3.25	28.91 ±0.01
3.50	28.88 ±0.03

4.50	28.79 ±0.02
5.50	28.68 ±0.02
6.00	28.61 ±0.03
7.50	28.46 ±0.03
8.00	28.39 ±0.01
8.50	28.35 ±0.01
9.50	28.26 ±0.02

Table 7-3 Resistance Variation with Radius of Curvature for Middle Finger Flex Sensor

Radius of Curvature (cm)	**Resistance (kΩ)**
1.50	18.25 ±0.01
2.50	17.94 ±0.015
3.25	17.75 ±0.02
3.50	17.66 ±0.02
4.50	17.41 ±0.02
5.50	17.17 ±0.02
6.00	17.02 ±0.03
7.50	16.65 ±0.03
8.00	16.48 ±0.014
8.50	16.34 ±0.05
9.50	16.15 ±0.02

Table 7-4 Resistance Variation with Radius of Curvature for Ring Finger Flex Sensor

Radius of Curvature (cm)	**Resistance (kΩ)**
1.50	8.26 ±0.01
2.50	8.09 ±0.03
3.25	7.94 ±0.01

3.50	7.89 ±0.01
4.50	7.71 ±0.02
5.50	7.57±0.03
6.00	7.47±0.03
7.50	7.29±0.04
8.00	7.23 ±0.03
8.50	7.19 ±0.01
9.50	7.10 ±0.02

Table 7-5 Resistance Variation with Radius of Curvature for Little Finger Flex Sensor

Radius of Curvature (cm)	**Resistance (kΩ)**
1.50	28.61 ±0.01
2.50	28.41 ±0.02
3.25	28.22 ±0.03
3.50	28.12 ±0.01
4.50	27.91 ±0.01
5.50	27.73 ±0.02
6.00	27.63 ±0.01
7.50	27.35 ±0.02
8.00	27.28 ±0.02
8.50	27.21 ±0.03
9.50	27.09 ±0.02

Similarly, Fig. 7.2, 7.3, 7.4, 7.5 gives the Resistance vs Radius of Curvature plots of the Index Finger Flex Sensor, Middle Finger Flex Sensor, Ring Finger Flex Sensor, Little Finger Flex Sensor respectively. From each plot, the sensitivity values of the individual sensors were obtained as follows: -0.11 kΩ/cm for Index Finger Flex Sensor, -0.26 kΩ/cm for Middle Finger Flex Sensor, -0.15 kΩ/cm for Ring Finger Flex Sensor, -0.19 kΩ/cm for Little Finger Flex Sensor.

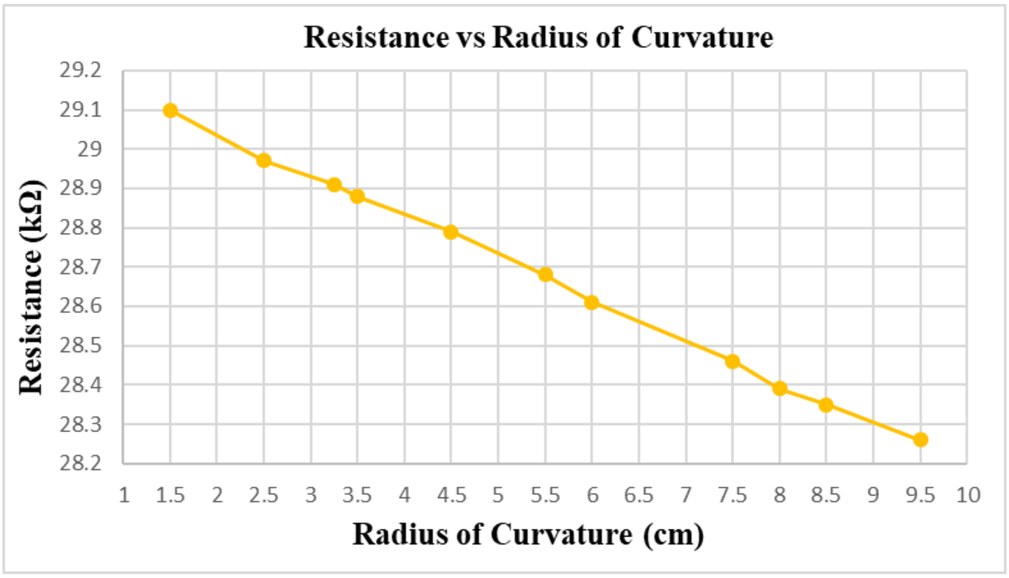


Fig. 7.2 Resistance vs Radius of Curvature plot for Index Finger Flex Sensor

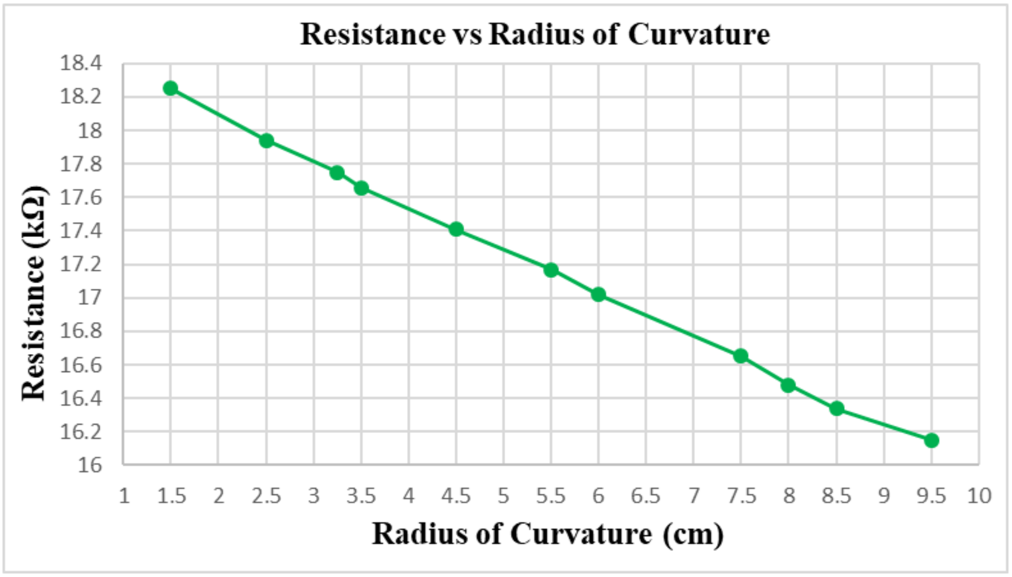


Fig. 7.3 Resistance vs Radius of Curvature plot for Middle Finger Flex Sensor

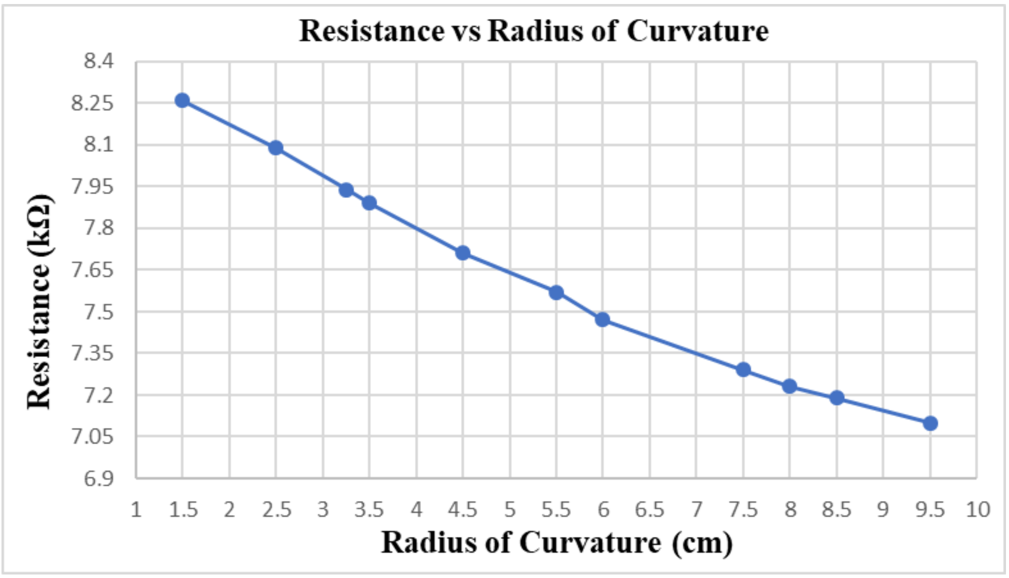


Fig. 7.4 Resistance vs Radius of Curvature plot for Ring Finger Flex Sensor

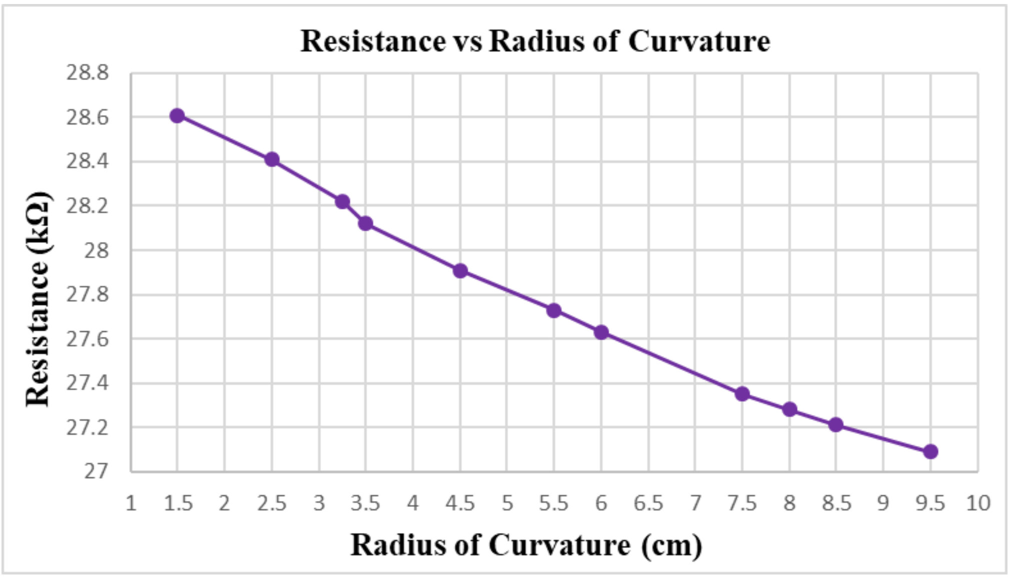


Fig. 7.5 Resistance vs Radius of Curvature plot for Little Finger Flex Sensor

7.2 Hardware Implementation

User can enable Mode 1 of operation of the Smart Hand-Wearable and access usage of common phrases like “Hello”, “Nice to meet you”, “Excuse Me”, “Thank You” etc. To activate Mode 1, the index finger is bent, and rest of the fingers are set upright to achieve the binary combination “01000” as shown in Fig. 7.6. The microcontroller determines the mode of operation as “MODE 1” and unlocks the phrases available in that mode. For instance, if someone greets “Good Morning”, the user can greet back “Good Morning” by bending the little finger and straightening out rest of the fingers which generates the combination “00001” and the phrase “Good Morning” is displayed and read out by the app as shown in Fig. 7.7.

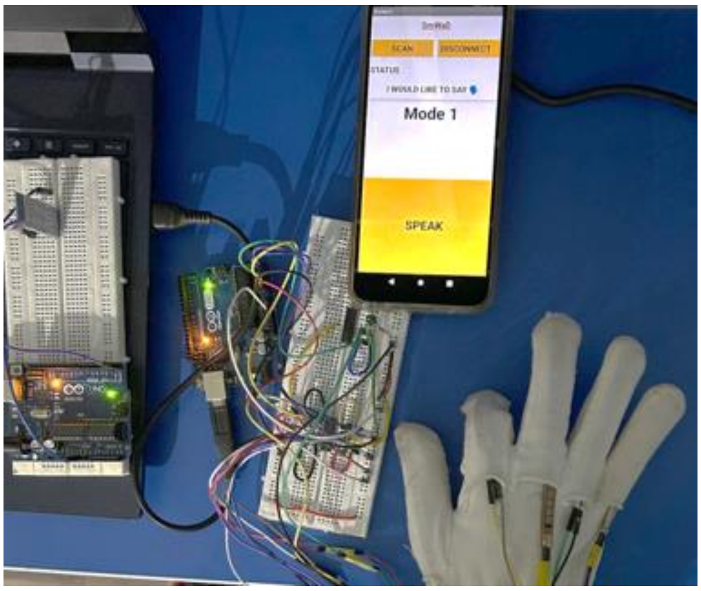


Fig. 7.6. Activation of Mode 1 by providing the input combination 01000.

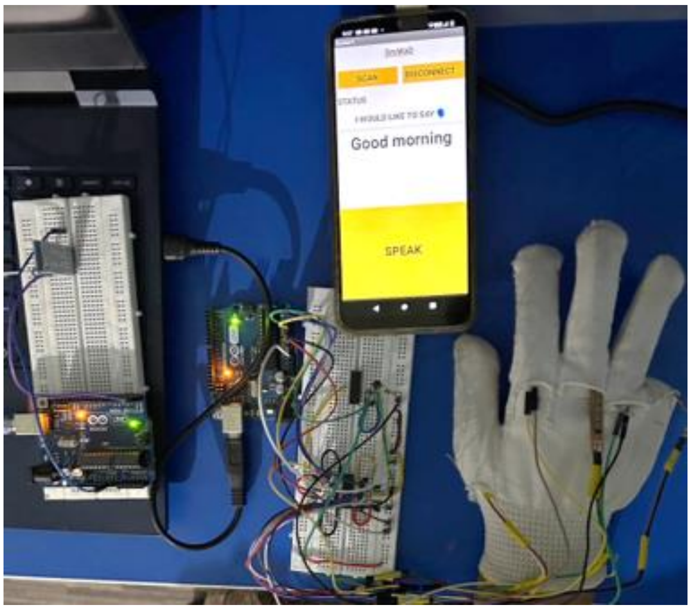


Fig. 7.7. Displaying “Good Morning” message by providing the combination 00001.

If asked for the user’s name, under Mode 1, user can make the combination “00101” by bending middle and little fingers, and the phrase “Hi! I am” is displayed and read out by the app. Further, user can enable Mode 2 of operation for accessing alphabets, by making the combination “01100”. Once mode 2 is activated, as shown in Fig. 7.8. user can spell out the name “XYZ” by making the combinations “11100”, “11101”, “11110” which are then displayed and read out by the app in a sequential fashion as “X Y Z”. Fig. 7.9.

shows the output "X' displayed in the screen of the application. Mode 3 of operation gives the user access to numbers and special characters. It is enabled by setting index, middle and ring fingers completely bent to achieve the combination "01110" as shown in Fig. 7.10. To prevent the overlapping of the phrases or alphabets, the user must exercise a small delay between two consecutive actions.

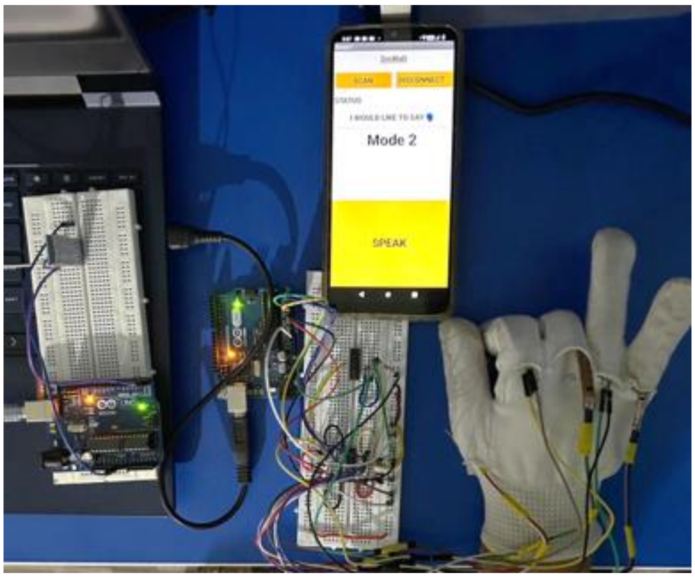


Fig. 7.8. Activation of Mode 2 by providing the input combination 01100.

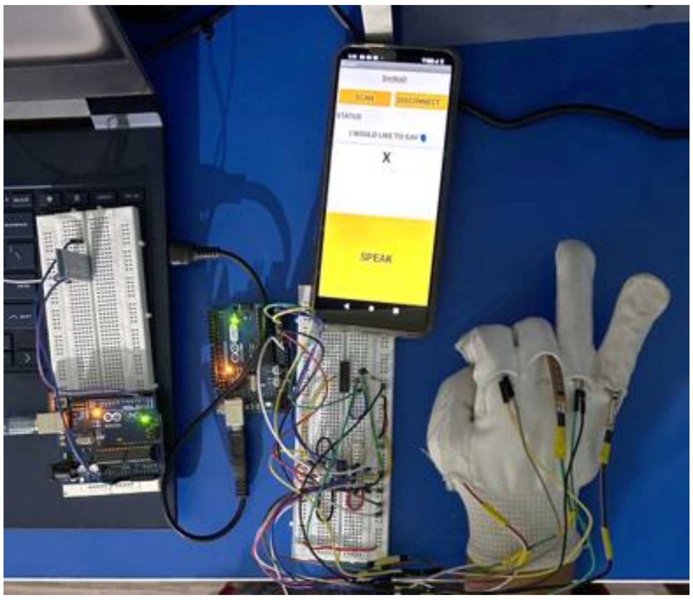


Fig. 7.9. Displaying "X" message by providing the input combination 11100.

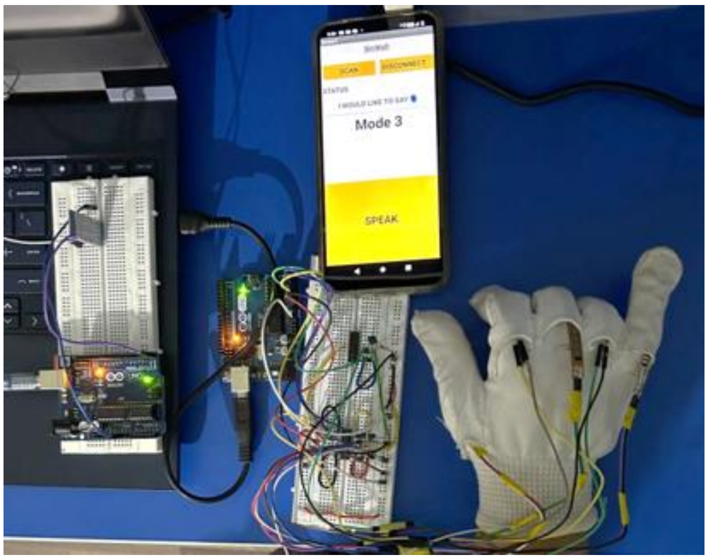


Fig. 7.10. Activation of Mode 3 by providing the input combination 01110.

7.3 Dataset

As mentioned in the report, a dataset was prepared by collecting finger-bending data from 100 individuals across age and gender using the smart hand wearable. The Data set is stored in Comma-separated values (CSV) Format. The sample of the stored sample is given in Table 5-6 and 5-7 gives the sensor reading values of 3 subjects for the combinations 11110 and 10100 respectively. Subject 1 is a 21-year-old male, subject 2 is 19-year-old male, subject 3 is a 57-year-old male.

Table 7-6 Sensor reading comparison between 3 subjects for binary combination 11110

	Thumb finger	**Index finger**	**Middle finger**	**Ring finger**	**Little finger**
Subjects	**Binary Combination**				
	1	**1**	**1**	**1**	**0**
21, Male	424	501	312	690	518
19, Male	426	503	326	712	486
57, Male	389	464	366	547	496

Table 7-7 Sensor reading comparison between 3 subjects for binary combination 10100

	Thumb finger	**Index finger**	**Middle finger**	**Ring finger**	**Little finger**
Subjects	**Binary Combination**				
	1	**0**	**1**	**0**	**0**
21, Male	418	508	293	692	525
19, Male	426	520	243	682	473
57, Male	395	469	373	560	507

In both tables, we can observe that the sensor readings of subject 3 when compared to subjects 1 and 2 has a huge variation. This is because as people age, major changes occur in their neuromuscular system. A reduction in hand strength, flexibility, and dexterity goes hand in hand with these alterations. Numerous investigations revealed that aged adults' force production tasks showed significant variability and decreased accuracy [9]. Due to these factors, subject 3's finger bending capability has decreased compared to subject 1 and 2.

7.3.1 Dataset Analysis

Statistical measures applied on the dataset, which includes mean, standard deviation, correlation matrix and feature importance gives us information on how well the values of flex sensor are correlated, indicating the level of accuracy we can expect.

(a) Mean and standard deviation

In table 7-8, the mean and standard deviation values of the flex sensors corresponding to 5 fingers are listed. It is clear that the values overlap to a certain degree and are not having much variation.

Table 7-8 Mean and Standard Deviation of flex sensor values for all 5 fingers

Metrics	**Thumb Finger**	**Index Finger**	**Middle Finger**	**Ring Finger**	**Little Finger**
Mean	384.25	437.53	386.66	630.29	454.75
Standard Deviation	24.61	52.37	69.24	51.27	40.52

(b) Correlation matrix

Even though the class distribution is equal, the correlation matrix shown in Fig. 7.11 indicates that each flex sensor output overlaps with one another in a greater degree. Index and thumb finger's flex sensor output overlaps significantly whereas flex sensor corresponding to middle finger has a least correlation with other sensor outputs. The overlap of flex sensor output corresponding to thumb finger with other sensor outputs are much higher, leading to errors in prediction of output.

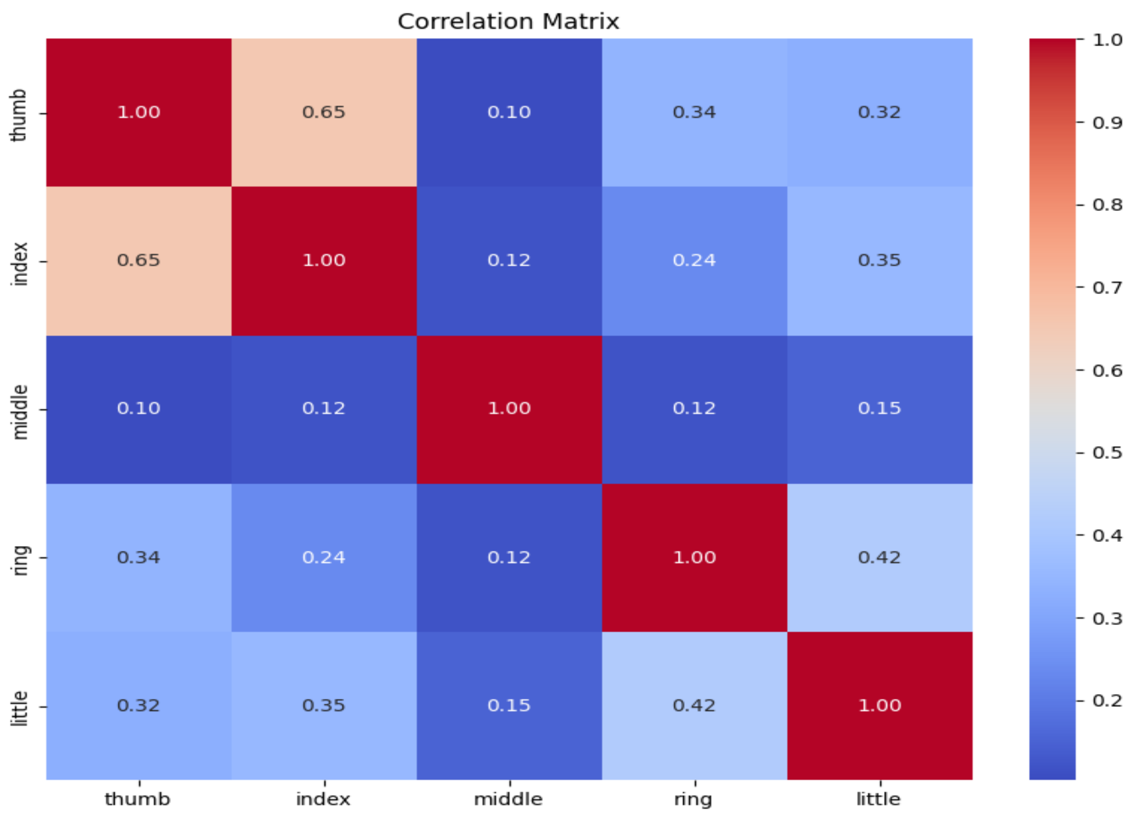


Fig. 7.11 Correlation matrix of the dataset

(c) Feature importance

When experimented with Random Forest model, as in Fig. 7.12, flex sensor output corresponding to middle finger has significant importance in prediction of output while flex sensor output corresponding to thumb finger.

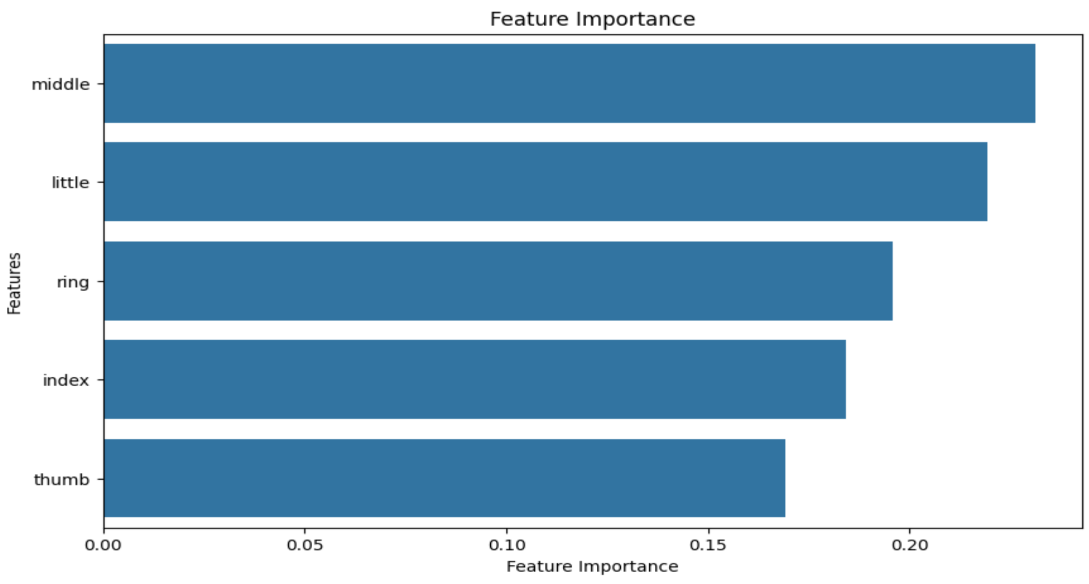


Fig. 7.12 Feature importance of each flex sensor output

7.3.2 Machine Learning Model Training and Evaluation

There are various machine learning models catering towards our need of increasing the accuracy and inference time. But there are 5 machine learning models chosen which provides better accuracy than others. Table 7-9 shows various metrics comparing the five chosen models, each having trade-off between accuracy and memory, while inference time for each model is suitable for real time applications. While simpler models like KNN classifier and Decision tree have a lower accuracy, the memory requirements are way lesser compared to top performing models like random forest, extra trees classifier and bagging classifier.

Table 7-9 Comparison of evaluation metrics for different machine learning models

Model	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1 Score (%)**	**Memory (MB)**	**Inference Time (ms)**
KNN Classifier	81.30	81.54	81.30	81.32	2.21	1.16
Decision Tree Classifier	80.60	80.69	80.60	80.60	3.57	0.08
Random Forest Classifier	90.17	90.21	90.00	90.17	669.80	11.30
Extra Trees Classifier	91.05	91.10	91.05	91.05	1100.00	11.95
Bagging Classifier	89.40	89.46	89.40	89.40	582.71	22.54

7.4 Performance Comparison with and without ML models

There are various parameters for cost-wise comparison and evaluate whether machine learning based system is a good choice. Table 7-10 shows the cost-wise and performance wise comparison.

Table 7-10 Cost and Performance comparison

System Type	**Monetary costs (Rs.)**	**Memory Requirements (GB)**	**Time Investment**	**Overall Accuracy (%)**
Without Machine Learning Classifier	3875	Negligible	Minimal (Rule-based setup)	79
With Machine Learning Classifier	3875	1	33.33 hours for dataset collection	91

Monetary costs are similar where the price of laptop used is neglected. The best performing model, Extra Trees Classifier, consumes 1 GB and as the previous instances are not stored, the memory requirement other than machine learning model is quite negligible. Time investment is calculated on the basis of hours required for the whole dataset collection. It takes 20 minutes per person, so for the total of 100 people, it takes 33.33 hours for complete data collection process. Both the glove systems are tested with 100 samples respectively. System embedded with machine learning got an overall accuracy of 91%, offering a significant improvement over the system without machine learning by a factor of 12%.

8. Conclusion

The proposed methodology aims to facilitate a new method of communication for speech impaired individuals. For sign language recognition, vision-based and sensor-based techniques are the most often used approaches. Traditional sensor-based devices for communication often suffer from complexity, cost, or inefficiency, such as gloves with predefined sentences. Our device enhances communication possibilities by introducing three distinct modes to a glove equipped with five flex sensors, simplifying design, and reducing complexity. By employing hybrid switching, users can communicate more easily, with the glove distinguishing hand gestures from predefined ones. This enhancement, alongside binary code allocation for gesture recognition, improves the glove's effectiveness and precision. Further, inclusion of machine learning algorithms, which were trained and tested on the dataset collected from a large diverse population, improved the gesture classification accuracy of the device, its robustness in diverse conditions and its compatibility with varying user needs and preferences. Ultimately, the smart hand-wearable empowers speech impaired individuals to lead more independent lives, overcoming their challenges and fostering growth. The proposed technique demonstrates efficiency, effectiveness, and precision, offering a promising solution for enhancing communication and autonomy for these individuals.

Bibliography

[1] A. Sahoo, G. Mishra, and K. Ravulakollu,"Sign language recognition: State of the art," *ARPN J. Eng. Appl. Sci*, vol. 9, no. 2, pp. 116–134, Feb. 2014.

[2] S. Baowidan, “Improving realism in automated fingerspelling of American sign language,” *Mach. Transl.*, vol. 35, no. 3, pp. 387–404, Sep. 2021.

[3] O. Al-Jarrah and A. Halawani, “Recognition of gestures in Arabic sign language using neuro-fuzzy systems,” *Artif. Intell.*, vol. 133, no. 1–2, pp. 117–138, Dec. 2001.

[4] M. J. Cheok, Z. Omar, and M. H. Jaward, “A review of hand gesture and sign language recognition techniques,” *Int. J. Mach. Learn. Cybern.*, vol. 10, no. 1, pp. 131–153, Jan. 2019.

[5] G. G. Poornima, A. Yaji, Achuth, A. M. Dsilva, and Chethana, “Review on text and speech conversion techniques based on hand gesture,” in *2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS)*, Madurai, India, 2021, pp. 1682-1689, doi: 10.1109/ICICCS51141.2021.9432277.

[6] I. S. M. Dissanayake, P. J. Wickramanayake, M. A. S. Mudunkotuwa, and P. W. N. Fernando, “Utalk: Sri Lankan sign language converter mobile app using image processing and machine learning,” in *2020 2nd International Conference on Advancements in Computing (ICAC)*, Malabe, Sri Lanka, 2020, pp. 31-36, doi: 10.1109/ICAC51239.2020.9357300.

[7] A. Zanzarukiya, B. Jethwa, M. Panchasara, and R. Parekh, “Assistive hand gesture glove for hearing and speech impaired,” in *2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184)*, Tirunelveli, India, 2020, pp. 837-841, doi: 10.1109/ICOEI48184.2020.9143031.

[8] K. Manikandan, A. Patidar, P. Walia, and A. B. Roy, ‘Hand gesture detection and conversion to speech and text’, *arXiv*, Nov. 2018, doi: 10.48550/arXiv.1811.11997.

[9] T. Yamunarani and G. Kanimozhi, ‘Hand gesture recognition system for disabled people using Arduino’, 2018. [Online]. Available: https://www.semanticscholar.org/paper/HAND-GESTURE-RECOGNITION-SYSTEM-FOR-DISABLED-PEOPLE-T.Yamunarani G.Kanimozhi/b1b3b388b8e70c96f170862c8cd6d3c0e9882a16

[10] S. Ghanem, C. Conly, and V. Athitsos, “A Survey on Sign Language Recognition Using Smartphones,” in *Proceedings of the 10th International Conference on Pervasive Technologies Related to Assistive Environments*, Island of Rhodes Greece: ACM, Jun. 2017, pp. 171–176, doi: 10.1145/3056540.3056549.

[11] J. J. A. Mendes Junior, M. L. B. Freitas, S. L. Stevan, and S. F. Pichorim, “Recognition of Libras Static Alphabet with MyoTM and Multi-Layer Perceptron,” in *XXVI Brazilian Congress on Biomedical Engineering*, vol. 70/2, R. Costa-Felix, J. C. Machado, and A. V. Alvarenga, Eds., in IFMBE Proceedings, vol. 70/2. , Singapore: Springer Singapore, 2019, pp. 413–419, doi: 10.1007/978-981-13-2517-5_63.

[12] J. Wu, L. Sun, and R. Jafari, “A wearable system for recognizing American sign language in real-time using IMU and surface EMG sensors,” *IEEE J. Biomed. Health Inform.*, vol. 20, no. 5, pp. 1281–1290, Sep. 2016.

[13] F. Wang, S. Zhao, X. Zhou, C. Li, M. Li, and Z. Zeng, “An recognition-verification mechanism for real-time Chinese sign language recognition based on multi-information fusion,” *Sensors (Basel)*, vol. 19, no. 11, pp. 2495, May 2019, doi: 10.3390/s19112495.

[14] H. Shaheen and T. Mehmood, “Talking gloves: Low-cost gesture recognition system for sign language translation,” in *2018 IEEE Region Ten Symposium (Tensymp)*, Sydney, Australia, 2018, pp. 219–224.

[15] R. Ambar, C. K. Fai, M. H. Abd Wahab, M. M. Abdul Jamil, and A. A. Ma’radzi, “Development of a wearable device for sign language recognition,” *J. Phys. Conf. Ser.*, vol. 1019, p. 012017, Jun. 2018.

[16] B. Ram, G. V, Chirranjeavi, Aaruran, and H. M, “Transhumeral sensory system to control robotic arm,” in *2023 IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER)*, Mangalore, India, 2023, pp. 185–190.

[17] J. Galka, M. Masior, M. Zaborski, and K. Barczewska, “Inertial motion sensing glove for sign language gesture acquisition and recognition,” *IEEE Sens. J.*, vol. 16, no. 16, pp. 6310–6316, Aug. 2016.

[18] N. T. Muralidharan, R. Ram, R. M., S. Nathan, and H. M., “Modelling of sign language smart glove based on bit equivalent implementation using flex sensor,” in *2022 International Conference on Wireless Communications Signal Processing and Networking (WiSPNET)*, Chennai, India, 2022, pp. 99–104.

[19] X. Huang et al., “Tracing the motion of finger joints for gesture recognition via sewing RGO-coated fibers onto a textile glove,” *IEEE Sens. J.*, vol. 19, no. 20, pp. 9504–9511, Oct. 2019.

[20] T. Kanokoda, Y. Kushitani, M. Shimada, and J.-I. Shirakashi, “Gesture prediction using wearable sensing systems with neural networks for temporal data analysis,” *Sensors (Basel)*, vol. 19, no. 3, pp. 710, Feb. 2019.

[21] G. Nirosha and R. Dr Velmani, “Raspberry Pi based sign to speech conversion system for mute community,” *IOP Conf. Ser. Mater. Sci. Eng.*, Dec. 2020, pp. 042005.

[22] K. A. Bhaskaran, A. G. Nair, K. D. Ram, K. Ananthanarayanan, and H. R. Nandi Vardhan, “Smart gloves for hand gesture recognition: Sign language to speech conversion system,” in *2016 International Conference on Robotics and Automation for Humanitarian Applications (RAHA)*, Kollam, 2016, pp. 1–6.

[23] B. Jose, “Smart Hand Glove for Hearing and Speech Impaired,” *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, pp. 978-984, Aug. 2021.

[24] A. Bansode, “Flex sensor based glove to control wheel chair and sign language translator for speech impaired people,” *SSRN Electron. J.*, 2020.

[25] A. M, A. S. Nair, A. Shreya, C. Tharun, and K. N. Neethu, “Design and implementation of smart gloves for the specially privileged,” in *2020 Third International Conference on Advances in Electronics, Computers and*

Communications (ICAECC), Bengaluru, India, 2020, pp. 1-5, doi: 10.1109/ICAECC50550.2020.9339475.

[26] Priiyadharshini, V. R. Balaji, Thrisha, and Suruthi, "Sign speaks-an IoT based smart gloves for dumb," in *2021 6th International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India, 2021.

[27] T. M. B. S. Balu, R. S. Raghav, K. Aravinth, M. Vamshi, M. E. Harikumar, and R. Gini, "Arduino based Automated Domestic Waste Segregator," in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, COIMBATORE, India, 2020.

[28] S. Bala Naga Pranav and M. Ganesan, "Plant signal extraction and analysis with the influence of sound waves," in *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, Coimbatore, India, 2020.

[29] B. G. Lee and S. M. Lee, "Smart wearable hand device for sign language interpretation system with sensors fusion," *IEEE Sens. J.*, vol. 18, no. 3, pp. 1224–1232, Feb. 2018.

[30] A. Suri, S. K. Singh, R. Sharma, P. Sharma, N. Garg, and R. Upadhyaya, "Development of Sign Language using Flex Sensors," in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, Trichy, India, 2020.

[31] J. Henderson, J. Condell, J. Connolly, D. Kelly, and K. Curran, "Review of wearable sensor-based health monitoring glove devices for Rheumatoid Arthritis," *Sensors (Basel)*, vol. 21, no. 5, pp. 1576, Feb. 2021.

[32] A. Babour, "Intelligent gloves: An IT intervention for deaf-mute people," *Journal of Intelligent Systems*, vol. 32, no. 1, 2023.

[33] T. S. Dias, J. J. A. Mendes Junior, and S. F. Pichorim, "Comparison between handcraft feature extraction and methods based on Recurrent Neural Network models for gesture recognition by instrumented gloves: A case for Brazilian Sign Language Alphabet," *Biomed. Signal Process. Control*, vol. 80, p. 104201, Feb. 2023.

[34] P. Telluri, S. Manam, S. Somarouthu, J. M. Oli, and C. Ramesh, "Low cost flex powered gesture detection system and its applications," in *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, Coimbatore, India, 2020.

[35] E. W. Thilanka and O. Patton, "Iot application development using mit app inventor to collect and analyze sensor data," in *2019 IEEE International Conference on Big Data (Big Data)*, IEEE, 2019, pp. 6157–6159.

[36] M. A. A. Faisal, Jun. 2022, "DU-ASL-DATA-GLOVE-DB," figshare, [Online]. Available: https://tinyurl.com/ydpuzsdw

[37] P. Sarma, H. K. Singh, and T. Bezboruah, "A Real-Time Data Acquisition System for Monitoring Sensor Data," *Int. J. Comput. Sci. Eng.*, vol. 6, no. 6, pp. 539–542, Jun. 2018.

[38] Y. Wang *et al.*, "Machine learning-enhanced flexible mechanical sensing," *Nanomicro Lett.*, vol. 15, no. 1, pp. 55, Feb. 2023.

[39] E. Y. Boateng, J. Otoo, and D. A. Abaye, "Basic Tenets of Classification Algorithms K Nearest-Neighbor, Support Vector Machine, Random Forest and

Neural Network: A Review," *Review. Journal of Data Analysis and Information Processing*, vol. 8, pp. 341–357, 2020.

[40] S. Zhang, "Challenges in KNN Classification," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4663–4675, Oct. 2022.

[41] R. Liu, L. Li, S. Pirasteh, Z. Lai, X. Yang, and H. Shahabi, "The performance quality of LR, SVM, and RF for earthquake-induced landslides susceptibility mapping incorporating remote sensing imagery,"*Arab. J. Geosci.*, vol. 14, no. 4, Feb. 2021.