Design and Study of Human-Human Interaction System to Control the Movement of Paralyzed Hands

Submitted by

Ms. Ankita Sanjiv Giroti

M. Sc. (Computer Science) Semester – IV (RP4 P02)

2025

Under the Guidance of

Dr. Mrs. A. D. Sakhare

Department of Electronics and Computer Science

Rashtrasant Tukadoji Maharaj Nagpur University Campus

Amravati Road, Nagpur – 440033

Certificate

This is to clarify that the project entitled **Design and Study of Human-Human Interaction System to Control the Movement of Paralyzed Hands** documents the participation of **Ms. Ankita Sanjiv Giroti**, under my supervision in the project program carried out during the semester IV of year 2025, in partial fulfilment of requirement prescribed for the degree of Master of Science, M. Sc. (Computer Science) of Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur.

Dr. Mrs. Anupama D. Sakhare

Project Guide and Head

Date:

Internal Examiner

External Examiner

Declaration

I, the undersigned, hereby declare that the project work entitled **Design and Study of Human-Human Interaction System to Control the Movement of Paralyzed Hands** submitted to the Rashtrasant Tukadoji Maharaj Nagpur University, Nagpur, is my independent work. This is my original work and has not been submitted anywhere for award of any degree/diploma. The system presented here has not been duplicated from any other sources. I understand that any such copying is liable to be punished in any way the University authority may deem fit.

Ms. Ankita S. Giroti

Place: Nagpur

Date:

Acknowledgement

I take this opportunity to express my gratitude towards my Project Supervisor and Head, **Dr. Mrs. Anupama Sakhare**, for her support throughout the completion of this project and without which I would have missed the experience and knowledge that I have gained from her. I take this opportunity to express my deep sense of gratitude and whole hearted thanks for her constant inspiration and guidance throughout the course of this project work.

I would like to express my deepest thanks to **Dr. Shivkumar Atre** for his guidance, support, and encouragement throughout the work. His direction has greatly helped me in my work.

I am thankful to **Dr. Mrs. Shweta Barhate** for her valuable co-operation and guidance during the project.

I am also very much thankful to **Dr. Mrs. Rakhi Vyaghra** and **Ms. Purva Manpe** for their necessary help, useful discussion and co-operation during the entire project work.

I am highly obliged to my parents for their consistent moral support and believing in me, my friends, other colleagues for their support. I would also like to express my thanks to my senior **Mr. Sankalp Bisan** for his valuable guidance and inspiration during the project work.

Lastly but not the least, I would like to thank all those who directly or indirectly helped me during my project work.

Date: Ms. Ankita Sanjiv Giroti

Index

Abstract

Monoplegia is the inability to move hand muscles due to nerve function loss, typically caused by a stroke or injury. This condition can severely limit independence and affect mental well-being. Traditional therapies often lack efficiency and can be time-consuming. The present work attempts to develop a rehabilitation system that assists paralyzed hands by mimicking healthy hand movements. EMG (electromyography) sensor is used to capture muscle activity from the unaffected hand and processed by an ESP32 WROOM microcontroller. The system consists of servo motors, PCA9685 Servo Motor Driver, and ESP32 WROOM to perform hand gestures. An ESP NOW protocol has been used to connect two ESP module via ad hoc network. The model uses TinyML (Tiny Machine Learning) technology is used to deploy Machine Learning model into microcontroller. K Nearest Neighbor (kNN) algorithm is used to train Machine Learning model. The present system captures real-time muscle signals, classifies them into open or closed hand gestures, and activates servos to reproduce these movements accurately.

Keywords: TinyML, EMG sensor, ESP NOW, Machine Learning (ML), k Nearest Neighbor, Servo Motor, Motor Driver

Chapter 1
Introduction

Chapter 1: Introduction

1.1 Background

Stroke is a life-threatening condition caused by the death of brain cells, often results in paralysis, affecting approximately 55% to 75% of survivors. The type and severity of paralysis can vary widely depending on the location of the brain damage. Similarly, spinal cord injuries can lead to hand paralysis, significantly impairing both physical function and emotional well-being. The extent of hand paralysis is influenced by the level of the spinal cord injury and the specific nerves affected. Both stroke and spinal cord injuries can have long-lasting effects on a person's quality of life, requiring ongoing rehabilitation and support. Physical therapy can improve muscle strength and help individual to regain motor functionality. The mirror therapy is the traditional method for improving hand motor function, in which mirror is placed between the affected and non-affected hand, and patient imagines the movement of non-affected hand by observing the movement of the affected hand. The rehabilitation system will help them to regain their muscle strength and improve their ability perform daily task.

1.2 Literature Review

An exhaustive review of literature has been performed and is discussed as follows:

Franceso Di Nardo et al.[1] studied the dataset of 2880 simulated sEMG signals for signal-to-noise ratio and time to train a hidden single layer fully connected neural network. DEMMAN's performance was used to evaluate sEMG signals and provide reliability prediction of muscle onset/offset in simulated and real sEMG signals. The present work predicted onset/offset timing of muscular recruitment of the machine learning based methods.

Ankita L. Dharmik et al.[2] designed an exoskeleton to assist a person with disability to perform routine tasks. EMG sensors were used to acquire the signals from the muscles of the hand and convert them into voltage and then fed to a signal conditioning circuit. Microcontroller was programmed to rotate servo motor according to the voltage received from the EMG sensors. The exoskeleton works effectively for normal functioning of the daily activities of an exoskeleton for hand rehabilitation.

Mailyn Calderon-Diaz et al.[3] discussed about the Hamstring strain injury, the most prevalent type of injury among professional soccer players. These injuries are challenging to pinpoint the most crucial risk. There is an increasing need for advanced injury detection and prediction models that can aid doctors in diagnosing or detecting injuries earlier and with greater accuracy. The study focuses on identifying biomarkers of muscle injuries through biomechanical analysis. The study employes several Machine Learning algorithms like Decision Tree, Discriminant method, Logistic Regression, Support Vector Machine (SVM), Artificial Neural Network (ANN), and XGBoost. The integration of machine learning models with fuzzy logic can be investigated to create a hybrid model and improve accuracy.

Ching Yee YONG and Terence Tien Lok SIA [4] Present a system that implemented Neuro-muscular Electrical Stimulator (NMES) integrated with Human-to-Human Interface (HHI) in the rehabilitation process for stroke patients from which a controller can control the motion of the subject by injecting his own signal to subject. The EMG detects the electrical signals transmitted from the motor neurons when there is muscle contraction. The signals were detected at the receiver's side by NMES and send to the actuators which helped subject to move their hand by mimicking the movements of the controller. The NMES is a device was used electrical impulses to activate muscle contraction. These impulses were delivered through electrode placed on the skin near targeted muscles.

Nestor J. Jarque-Bou et al.[5] reviewed the studies of EMG to characterize the muscle activity of the forearm and hand. The paper focused on human hand behavior, improve rehabilitation, control of prostheses, and biomechanical models. Myoelectric hand prostheses used the electrical action potential of the residual muscles in the limb emitted during muscle contractions. The methods like detection, decomposition, processing, and classifications were used to acquire EMG signals.

Yassine Bouteraa et al.[6] presented a remote rehabilitation system that integrates a IoT based connected robot for wrist and forearm rehabilitation which uses Support Vector Machine (SVM) to classify the design for the estimation of muscle fatigue based on the features extracted from the EMG signal acquire from the patient. The rehabilitation robot integrates the biomechanics of the forearm and wrist joints into rehabilitation protocol. The SVM classifier was used to estimate muscle strain based on feature extracted from the EMG signal acquired from the patient.

Md. Latifur Rahman et al.[7] presented the design of a low-cost muscle stimulator integrated with IoT to make it more user friendly. The feedback system was developed to receive voltage and send to the IoT device which could be used to determine muscle strength. The IoT device then sends data to a cloud server and stored against the ID of each user so that physiotherapist can monitor the patient. The advantage of this system is that it's affordable, so everyone can use it without spending a lot on hospital bills.

Nat Wannawas and Aldo Faisal [8] presented a method which uses Reinforcement Learning to observe muscle fatigue. The method is based on Gaussain State-space Model (GSSM) that utilizes the Recurrent Neural Network (RNN). The GSSM functions filter that converts an observable environment into a state representation vector. Reinforcement Learning learns a task through reward signals collected from interactions with an environment., occurs in a discrete time. RL-GSSM was trained to control arm movements through muscle stimulation under progressive muscular fatigue.

Dr. G. Sophia Reena [9] discussed the design, development, and testing of a tiny wearable interface that uses an array of sensors to monitor and notify a person's back position in real time. The device was capable of detecting and correcting improper posture using wearable EMG and flex sensor. Tiny Machine Learning to monitor human posture and categorize postures in real time. The data fed to the Arduino UNO for further processing of the signal. Flex sensor was used for deflection or blending of the body and EMG sensors record the electrical activity produced by skeletal muscles. LCD was used to display improper posture to the user.

DERVİŞ PAŞA [10] developed a stimulator for drop foot syndrome where a person faces difficulty in lifting the front part of the foot. Functional Electrical Stimulator (FES) uses Force sensitive Resistors (FSR), programmable microcontroller, transmitter, receiver, and electrodes. The system helped many patients with foot drop problem to move easily and comfortably. It worked by supplying electrical pulses to the nervous system to stimulate paralyzed muscles by producing muscular contractions.

Xiaoshi Chen et al.[11] Present the human-machine interaction system based on the concept of mirror therapy. Surface EMG was used to collect data from non-affected hand which classify the hand gestures using machine learning model. The STM32F103 was used to integrate DC motor and other sensors. Flex sensor and pressure sensors were used to process the signals coming from the sEMG. Machine Learning algorithms like Support Vector Machine (SVM), k-NN, and subspace k-NN were used for pattern recognition.

Xiaoshi Chen et al.[12] designed a tendon-driven motor glove driven by linear actuator powered by DC micro motor. Flexion sensors and force sensors are also mounted on each finger. Wi-Fi interface was implemented to send and receive control signals. The paper is the continuous work of the previous paper.

Alireza Mohammadi et al.[13] designed the portable tendon-driven soft exoskeleton glove for hand rehabilitation and assistance. Thermoplastic polyurethane (TPU) was used for a 3D printing of the exoskeleton. DC motors were implemented on each finger which was controlled using ATmega2500 microcontroller. The pinch, cylindrical grasp force, and maximum grasp size were measured using two flat surface resistors and two force surface resistors.

Waqas Ahmed et al.[14] developed a robotic hand controlled by the soft glove placed on the healthy hand. The goal was to perform mirror therapy on the robotic hand. Two experiments were performed, the one was feature extraction and gesture recognition, and other was real-time performance for sensing actuation combination of the glove. The output of the flex sensor was evaluated using Arduino and sent it to servo motor to perform the same action.

Xuanyi Zhou et al.[15] developed a tendon-driven robotic finger using servo motors. The robotic finger was controlled by a soft glove worn on a human hand, consist of 14 pressure sensors and 5 bend sensors to collect the data of a human hand. The finger consists of a base, servo drive wheels, and servo motors placed on a base far away from the finger joints.

1.3 Problem Statement and Definition

Stroke is a life-threatening condition that can lead to loss of motor function, affecting various parts of the body, including the face, arms, and legs. The loss of hand function, in particular, can significantly hinder daily activities and have a devastating emotional impact. However, with appropriate rehabilitation, many individuals can regain some or all of their lost motor function. The present system aims to assist in regaining hand functionality by facilitating fine-motor activities. The system will mimic the movements of the unaffected hand.

1.4 Significance of Present Work

The present rehabilitation system aims to assist individuals with hand paralysis in regaining motor functionality, enabling them to perform daily activities more independently. By mirroring the

movements of the unaffected hand, the system offers a novel approach to rehabilitation, potentially improving patient outcomes and offering a more effective rehabilitation system.

1.5 Objectives of Present Work

The objectives of the present system are as follows:

- To design a rehabilitation system for an individual with hand paralysis to regain functionality of their hand
- To design a system that senses electromyography (EMG) signals of the hands of the therapist
- To train a machine learning model to recognize hand movements
- To mimic the hand movement of the therapist to increase flexibility of the patient's hand

1.6 Orientation of the Present Work

The present rehabilitation system is design to assist individuals with hand paralysis caused by injuries or stroke. The system consists of two primary components: a sensor phase and an actuator phase. The sensor phase utilizes an electromyography (EMG) sensor connected to a microcontroller on the unaffected hand to capture muscle activity. The actuator phase employs servo motors and a microcontroller to control the movements of the paralyzed hand, mirroring the actions of the unaffected hand. To communicate between two microcontrollers, an ad-hoc network ESP NOW is used.

Chapter 2

Requirements and Analysis

Chapter 2: Requirements and Analysis

2.1 Requirements Specification

2.1.1 Hardware

1. **Computer**

Processor: Intel Core i5-9400 processor with 2.90GHz clock speed

System: Windows 10 with 64-bit operating system

RAM: 4 GB

Hard Disk: 500 GB

2. **ESP32 WROOM**

ESP32 WROOM is a low-cost, low-power System on a Chip (SoC) microcontrollers developed by Espressif. It includes Wi-Fi and Bluetooth wireless capabilities and dual-core processor. It supports low-power mode states like deep sleep to save power. Easily get connected to Wi-Fi network over the Internet. It has 2 Xtensa 32-bit LX6 microprocessors and wide variety of input and output peripherals. It also supports Bluetooth and BLE (Bluetooth Low Energy)

Specifications:

Wi-Fi: 150.0 Mbps data rate with HT40

BLE (Bluetooth Low Energy) and Bluetooth Classic

Tensilica Xtensa Dual-Core 32-bit LX6 microprocessor

Speed ranges from 160 to 240 MHz

ROM: 448 KB (for booting and core functions)

SRAM: 520 KB

Low power consumption

Fig. 2.1: Arduino UNO

3. **EMG Sensor**

EMG Muscle Sensor Module V3 is used to measure the muscle activity of the human body. It has been traditionally been used for medical research and diagnosis of neuromuscular disorders. The sensor is specifically designed for microcontrollers and it also includes adjustable gain to improve ruggedness of the signal.

Specification:

Dual power supply of 9 volts

In-built adjustable gain

Signal pin to directly communicate with the controller

Fig. 2.2: EMG Sensor

4. **Servo Motor**

The servo motor SG90 is used to control the linear motion of the object in a 180^{o} direction. It consists of three wires brown. Red, and orange for ground, +5V, and PWM signal, respectively. The servo motors can be implemented of the device to control the movements of the paralyzed hand.

Fig. 2.3: Servo Motor

5. PCA9685 Servo Motor Driver

PCA9685 module is a 16 Channel Servo Motor Driver used to control upto 16 servo motors with just two pins of the ESP32 microcontroller with I2C. Controlling the servos is simple and easy. Servos can be plugged in directly to the board with no additional circuits or connectors required.

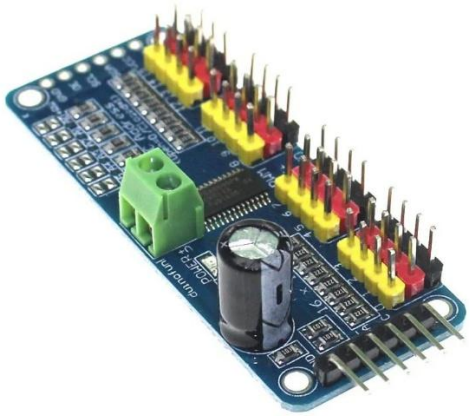


Fig. 2.4: PCA9685 Servo Motor Driver

2.1.2 Software

1. **Arduino IDE**

   Arduino IDE is a cross-platform open-source application that allows users to write, compile, debug, and upload code to Arduino boards. It contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them. The console displays text output by the Arduino Software (IDE), including complete error messages and other information.

2. **Spyder IDE**

   Spyder is a free and open-source scientific Python integrated development environment (IDE) for scientific computing, data science, and machine learning. It is written in Python, Qt, and PyQt, and is designed to be a lightweight, cross-platform IDE that is easy to use and extend. Spyder is a good choice for both beginners and experienced Python developers, as it is easy to use and extend. It includes a variety of features that make it well-suited for scientific computing, such as a powerful code editor, a console for interactive Python execution, a variable explorer, a debugger, and a profiler.

2.1.3 Libraries

1. **NumPy:** NumPy (Numerical Python) is a python library intended for computation. It aims to provide an array object known as **ndarray.** NumPy arrays are stored at one continuous place in memory, so that processes can access and manipulate them very efficiently.
2. **Pandas:** It is a Python library used to work with datasets for analyzing, cleaning, exploring, and manipulating data. It provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data in a easy and intuitive way. It aims to be the fundamental high-level building block for real world data analysis in Python. It is a most powerful and flexible open source data analysis and manipulation tool available in any language. Relevant data is very important to train the ML model, Pandas can be used to clean the messy data in the dataset.

3. **Scikit-learn:** Scikit-learn is a powerful and open- source machine learning library. It provides various tools for model fitting, data preprocessing, model selection, model evaluation, and many other utilities. It also provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistence interface in Python. The library is highly written in Python but built upon NumPy, SciPy and Matplotlib.
4. **Keras:** Keras is an open-source library that runs efficiently on CPU as well as GPU. It is used for deep learning. The popular ML library works with the building blocks of neural networks such as activation functions, layers, objectives, and optimizers.
5. **Tensorflow:** TensorFlow is an open-source library for numerical computation, large-scale machine learning, deep learning, and other statistical and predictive analytics workloads. TensorFlow is developed by Google for deep learning applications. It supports traditional machine learning and accepts data in multi-dimensional arrays of higher dimensions called tensors. It combines various machine learning and deep learning models and algorithms, and makes them useful by way of a common interface. It offers APIs for both Python and C++. It is designed to streamline the process of developing and executing advanced analytics applications for users such as data scientists, statisticians, and predictive modelers.
6. **Matplotlib:** Matplotlib is a powerful and versatile open-source plotting library for Python, designed to help users visualize data in a variety of formats. It is used to create static, animated, and interactive visualizations in Python. It makes easy things easy and hard things possible.
7. **Serial:** The serial library in Python facilitates communication with devices connected via serial ports. It enables sending and receiving data between a computer and hardware like microcontrollers, sensors, and other peripherals.

2.2 Planning and Scheduling

- Understanding the problem domain and the difficulties people facing with hand paralysis
- Gathering information related to the specified problem
- Collection of the required hardware components and software applications are specified in the requirement specification
- Analyzing the feasibility of the project
- Study of the suitable Machine Learning model that can be deployed on the system
- Designing the Present system and verifying whether it is accepting inputs
- Collecting or creating dataset
- Training the Machine Learning model using suitable ML algorithm
- Deploying the ML model on the designed system
- Analyzing the results obtained by giving real time input to the system
- The system will recognize the gesture and perform the same action using servo motor

The present system will follow an iterative model for the development of the system. The iterative model is a type of software development life cycle model that focuses on initial, basic implementation that gradually adds more complexity to the system. Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. It break down the software development of a massive program into smaller components. It typically involves data collection, training ML model, evaluating, improving, and repeating the process until the model is performing well. The workflow of the iterative model is given below:

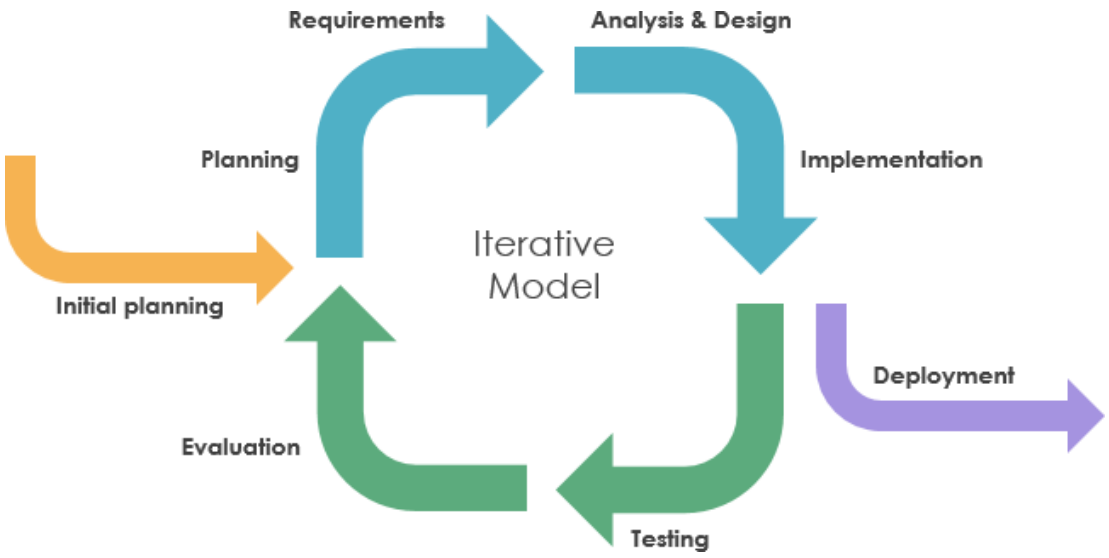


Fig. 2.5: Iterative Model

[Source: medium.com]

- **Requirements Gathering and Analysis:** In this phase, gathering and analyzing of the requirements for the system takes place. This includes identifying the needs of the users and stakeholders, and determining the features and functionality that the system must have.
- **Planning:** The team creates a plan for the development of the system. This includes estimating the time and resources required, and identifying the risks and challenges that the team may face.
- **Design:** The team designs the system architecture and user interface. This includes identifying the main components of the system and how they will interact with each other, and designing the screens and menus that users will see.
- **Implementation:** The team implements the system in small, incremental steps. Each increment should be a working product that can be tested and deployed to users.
- **Testing:** The team tests each increment of the system to ensure that it meets the requirements and that it is free of defects.
- **Deployment:** The team deploys the system to users and provides support for the system.
- **Maintenance:** The team maintains the system by fixing bugs and adding new features and functionality as needed.

2.3 Methodologies

The development of the present work will be done as shown in the figure below:

2.3.1 Experimental Technique

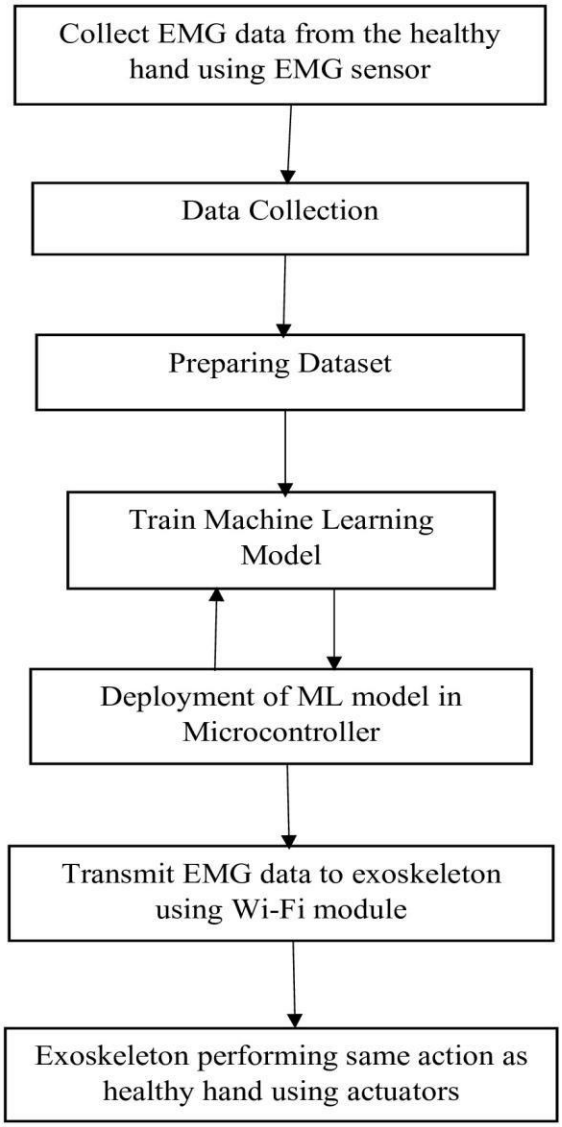


Fig. 2.6: Development of the Present System

Fig. 2.6. explains the work flow of the system. The system will take the real-time input will be taken from the healthy hand using EMG sensor. A machine learning model is prepared and trained to detect patterns in the collected data. The trained model is then deployed onto a microcontroller within the exoskeleton. Real-time EMG data from the healthy hand is transmitted to the exoskeleton via Wi-Fi, where it is processed by the model to control the exoskeleton's actuators, allowing it to mimic the movements of the healthy hand.

2.3.2 Input

The input of the system is the gesture performed by the hand through EMG sensor in real time.

2.3.3 Dataset

The dataset has been collected from the healthy hand using EMG sensors for open and closed hand gestures. The dataset contains raw values of the EMG ranging from 500 to 2000. Total of 10000 samples are collects from one hand to process the data into a csv file.

2.3.4 Algorithms

Artificial Neural Network (ANN)

An Artificial Neural Network (ANN) is a computational model inspired by the way biological neural networks in the human brain process information. ANNs are a key technology in machine learning and artificial intelligence, used for a variety of applications such as classification, regression, clustering, and more. It consists of neurons, similar to human brain cells, which receive input, process it, and produce output. Every neuron has an associated weight with it which can be adjust during the learning process. Neurons are organized into three distinct layers: Input layer, Hidden layer, and Output layer. An activation function is applied to the output of each neuron to introduce non-linearity to the model. Neural network can handle complex tasks that involve large datasets, complex pattern recognition, and predictive capabilities.

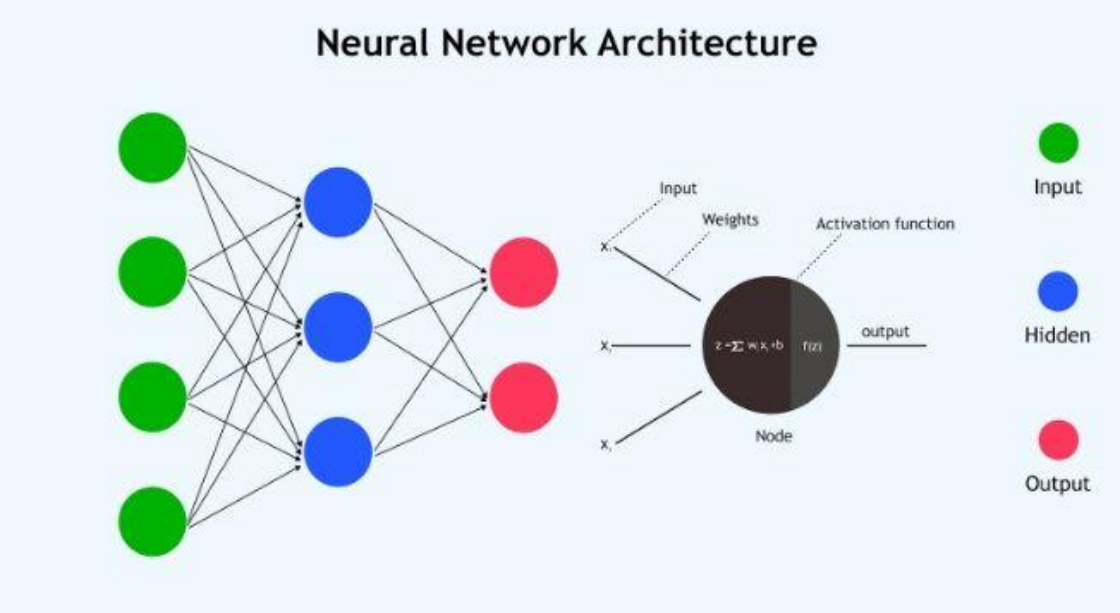


Fig. 2.7: ANN Architecture [upgrad.com]

2.4 Alternative Technique

Long-Short Term Memory (LSTM)

Long Short-Term Memory is a type of Recurrent Neural Network that can keep long-term dependencies in sequential data. LSTMs can process and analyze sequential data, such as time series, text, and speech. They use a memory cell and gates to control the flow of information, allowing them to selectively retain or discard information as needed. There are three types of gates in an LSTM: the input gate, the forget gate, and the output gate, implemented using sigmoid functions, which produce an output between 0 and 1. The gates in an LSTM are trained to open and close based on the input and the previous hidden state. This allows the LSTM to selectively retain or discard information, making it more effective at capturing long-term dependencies.

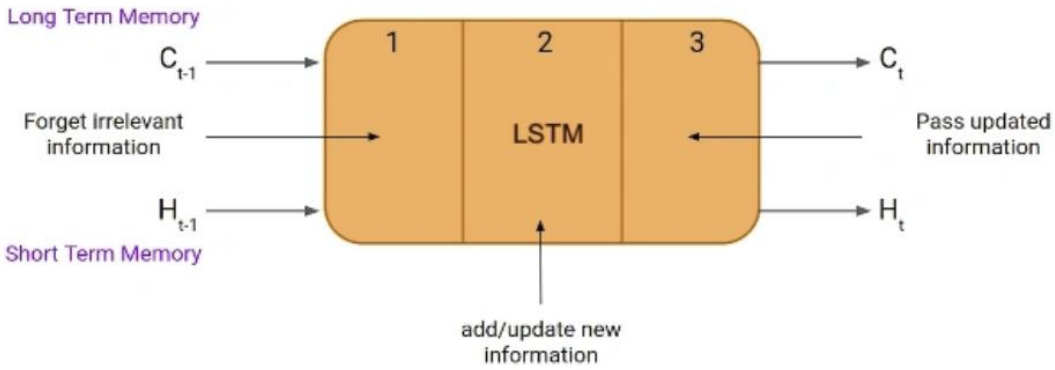


Fig. 2.8: Long Short Term Memory (LSTM) [analyticsvidhya.com]

Support Vector Machine (SVM)

Support Vector Machine is a supervised machine learning technique that classifies data by finding an optimal line or hyperplane to maximizes the distance between each class in an N-dimensional space. It can work for linear as well as non-linear classification, whereas for non-linear data, kernel functions are used to transform the high-dimension space to perform linear separation. The aim is to find the decision boundary that separates the data points of different classes.

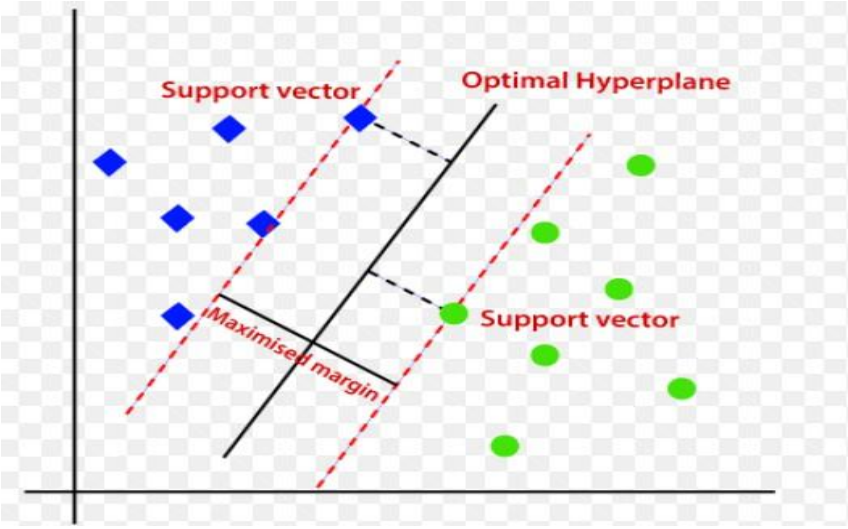


Fig. 2.9: Support Vector Machine [Javatpoint.com]

Chapter 3

System Design

Chapter 3: System Design

3.1 Block Diagram

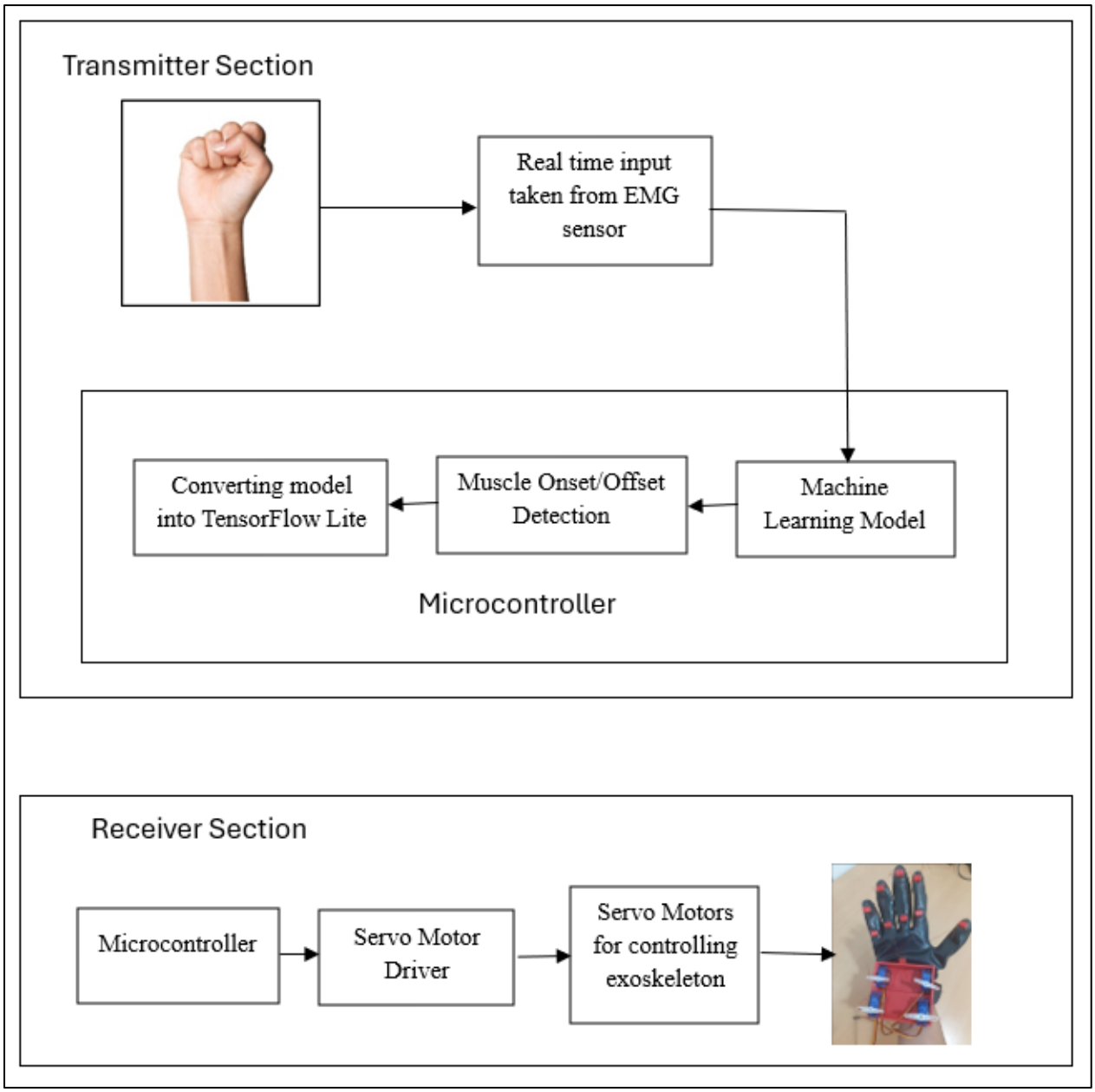


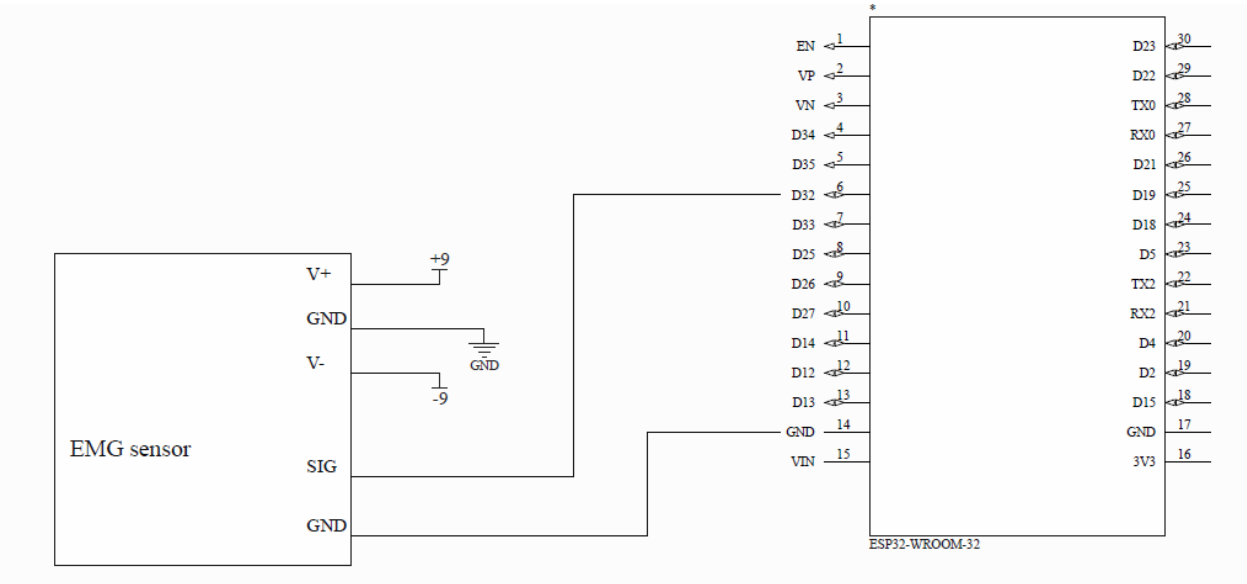
Fig 3.1: Block Diagram of Present Work

The block diagram of the present work is shown in the Fig. 3.1. It consists of EMG sensor captures real-time muscle signals from the healthy hand. These signals are pre-processed and then input into a machine learning model running on a microcontroller, classifies a hand gesture as open or

closed. The classification result is sent to a second microcontroller, which controls servo motors on a wearable glove to replicate the same gesture on the affected hand.

3.2 Circuit Diagram

Fig. 3.2 depicts the circuit diagram of the present work. The circuit diagram uses two ESP32 WROOM module which are communicating through ESP NOW protocol. The first ESP32 collects EMG signals from the healthy hand via a three-lead Muscle V3 sensor, which features five pins: three for dual-mode power supply, one for analog-to-digital signal output, and one ground connection to the microcontroller. The processed EMG data is then transmitted to the second ESP32 module using ESP-NOW. The second ESP32 is responsible for actuating servo motors mounted on the wearable glove. For motor control, an Adafruit PCA9685 16-channel, 12-bit PWM/Servo Driver with an I2C interface is employed: its SDA and SCL pins are connected to GPIO21 and GPIO22 of the ESP32, respectively, while VCC and GND are connected to the 3.3V and ground pins of the ESP32. Additionally, 2.7kΩ pull-up resistors are connected from the SDA and SCL lines to VCC to ensure reliable I2C communication. PWM channels 0 to 4 of the PCA9685 are each connected to one of the five servo motors on the glove.



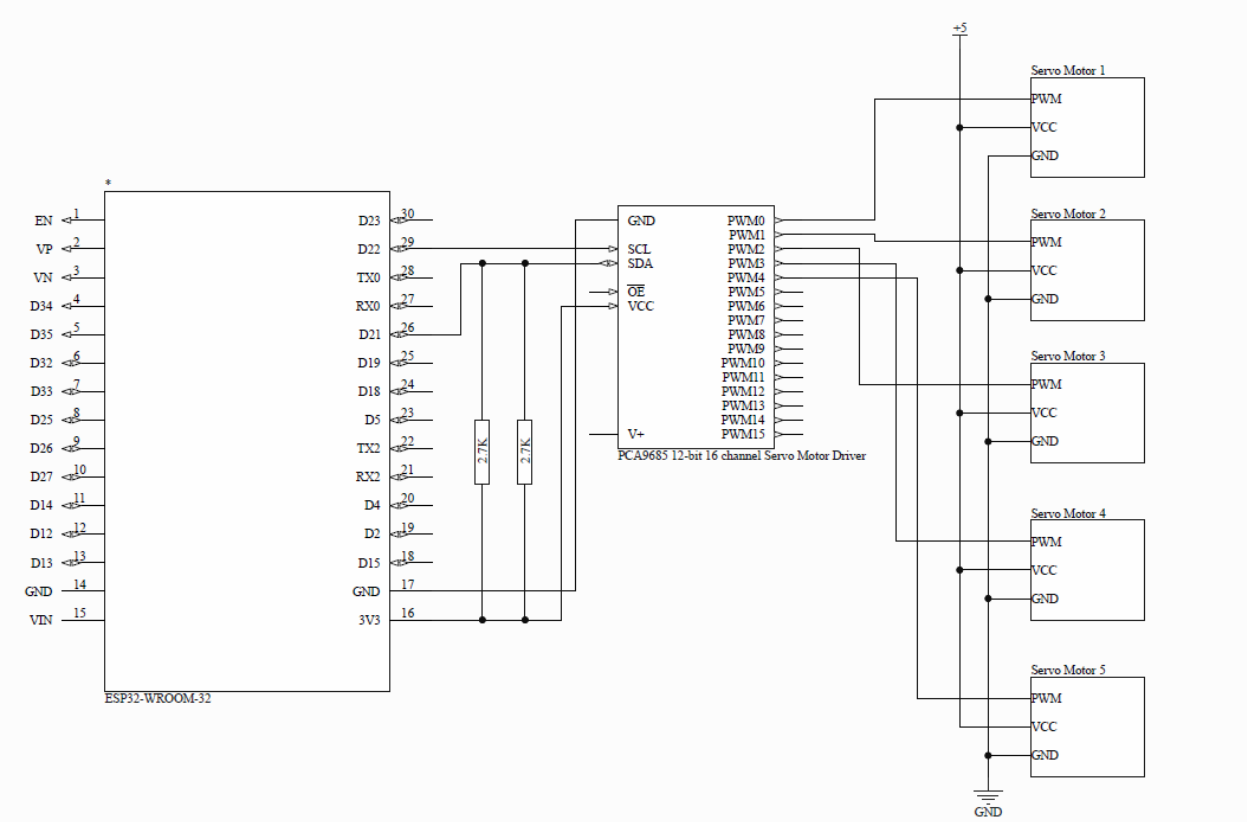


Fig 3.2: Circuit Diagram of the Present System

3.3 System Physical Architecture

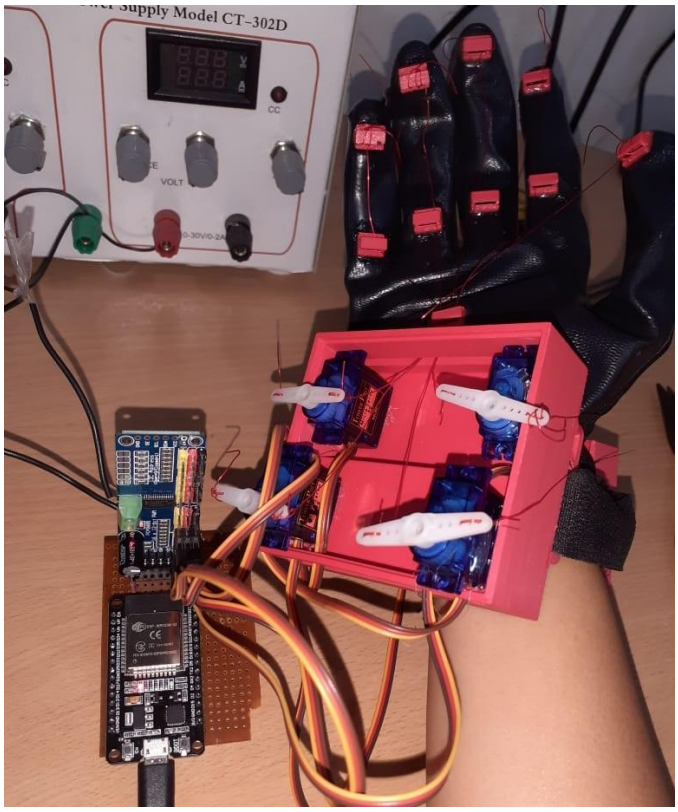


Fig. 3.3: Exoskeleton with mounted servo motors

Chapter 4

Experimental Work

Chapter 4: Experimental Work

4.1 Coding Details

Arduino Files:

1. Read muscle data using EMG Sensor:

```
#include <esp_now.h>
#include <WiFi.h>
#define EMG_PIN 32

uint8_t sendDataToMacAddr[] = {0xE8, 0x6B, 0xEA, 0xDF, 0x4A, 0x00};
typedef struct emg_signal {
  int value;
} emg_signal;
emg_signal mySignal;
esp_now_peer_info_t peerInfo;

void onDataSend(const uint8_t *macaddr, esp_now_send_status_t sendStatus) {
  Serial.print("Delivery Status: ");
  Serial.println(sendStatus == ESP_NOW_SEND_SUCCESS ? "Delivery Successeful": "Delivery
Fail");
}

void setup() {
  Serial.begin(9600);
  WiFi.mode(WIFI_STA);
  if (esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
```

```
  esp_now_register_send_cb(onDataSend);
  memcpy(peerInfo.peer_addr, sendDataToMacAddr, 6);
  peerInfo.channel = 0;
  peerInfo.encrypt = false;

   if (esp_now_add_peer(&peerInfo) != ESP_OK) {
    Serial.println("Failed to add peer");
    return;
  }
}

void loop() {
  mySignal.value = analogRead(EMG_PIN);
  Serial.println(mySignal.value);
  esp_now_send(sendDataToMacAddr, (uint8_t *) &mySignal, sizeof(mySignal));
  delay(400);
}
```

2. Rotate servo motors:

```
#include <Wire.h>
#include <WiFi.h>
#include <Adafruit_PWMServoDriver.h>
#include <esp_now.h>
#define THRESHOLD 1000

Adafruit_PWMServoDriver pca9685 = Adafruit_PWMServoDriver(0x40);

#define SERVOMIN  200  // Minimum value
#define SERVOMAX  600  // Maximum value
```

```
#define SER0  0   //Servo Motor 0 on connector 0
#define SER1  1  //Servo Motor 1 on connector 1
#define SER2  2  //Servo Motor 1 on connector 2
#define SER3  3  //Servo Motor 1 on connector 3
#define SER4  4  //Servo Motor 1 on connector 4

// Variables for Servo Motor positions
int pwm0;
int pwm1;
int pwm2;
int pwm3;
int pwm4;

typedef struct receive_signal {
  int value;
} receive_signal;

receive_signal mySignal = {0};

void onDataRecv(const esp_now_recv_info *mac, const uint8_t *incomingData, int len)
{
  if(len == sizeof(mySignal)) {
    memcpy(&mySignal, incomingData, sizeof(mySignal));
    Serial.println(mySignal.value);
  } else {
    Serial.println("Received data length mismatch");
  }
}
```

```
void setup() {
  Serial.begin(9600);
  pca9685.begin();
  pca9685.setPWMFreq(50);
  WiFi.mode(WIFI_STA);
  if(esp_now_init() != 0) {
    Serial.println("Error initializing ESP-NOW");
    return;
  }
  esp_now_register_recv_cb(onDataRecv);
}

void loop() {
  if(mySignal.value > THRESHOLD) {
     // Calculate PWM for all servos
    int pwm0 = map(180, 0, 180, SERVOMIN, SERVOMAX);
    int pwm1 = map(180, 0, 180, SERVOMIN, SERVOMAX);
    int pwm2 = map(180, 0, 180, SERVOMIN, SERVOMAX);
    int pwm3 = map(180, 0, 180, SERVOMIN, SERVOMAX);
    int pwm4 = map(180, 0, 180, SERVOMIN, SERVOMAX);

     // Update all servos simultaneously
    pca9685.setPWM(SER0, 180, pwm0);
    pca9685.setPWM(SER1, 180, pwm1);
    pca9685.setPWM(SER2, 180, pwm2);
    pca9685.setPWM(SER3, 180, pwm3);
    pca9685.setPWM(SER4, 180, pwm4);
  }
  else if(mySignal.value < THRESHOLD) {
```

```
  // Calculate PWM for all servos
  int pwm0 = map(0, 0, 180, SERVOMIN, SERVOMAX);
  int pwm1 = map(0, 0, 180, SERVOMIN, SERVOMAX);
  int pwm2 = map(0, 0, 180, SERVOMIN, SERVOMAX);
  int pwm3 = map(0, 0, 180, SERVOMIN, SERVOMAX);
  int pwm4 = map(0, 0, 180, SERVOMIN, SERVOMAX);

  // Update all servos simultaneously
  pca9685.setPWM(SER0, 0, pwm0);
  pca9685.setPWM(SER1, 0, pwm1);
  pca9685.setPWM(SER2, 0, pwm2);
  pca9685.setPWM(SER3, 0, pwm3);
  pca9685.setPWM(SER4, 0, pwm4);
 }
}
```

Output:

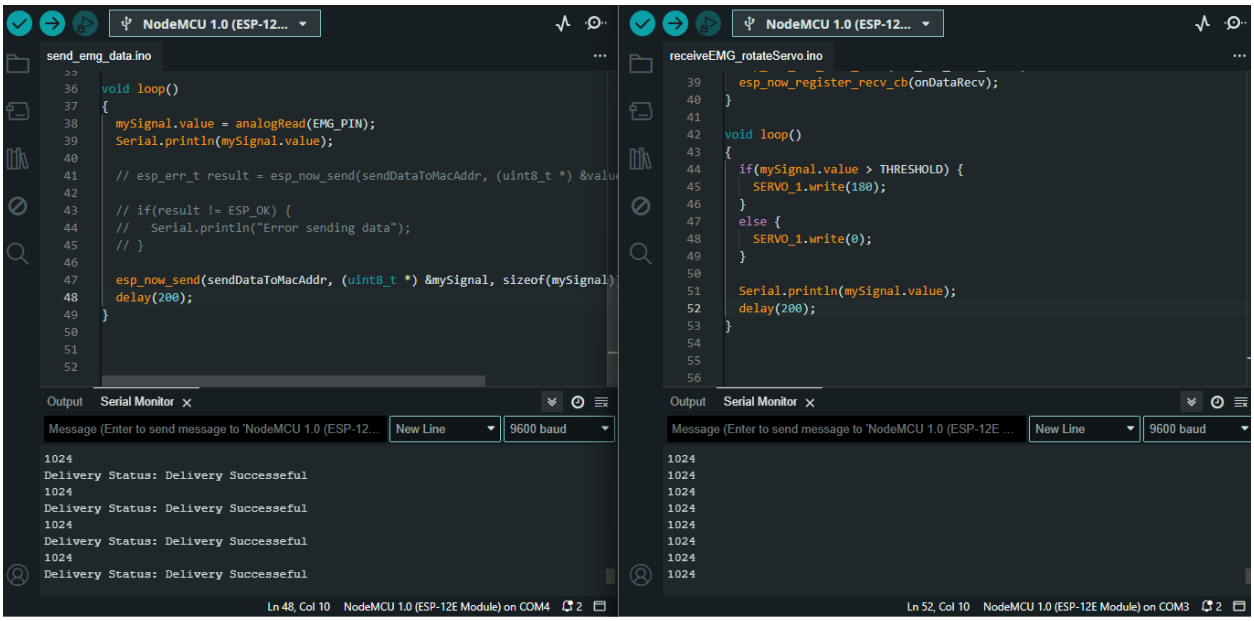


Fig. 4.1: Data received at servo successfully from EMG sensor

Python Files:

1. Dataset Collection

```
import serial
import csv
import time

esp_port = 'COM3'
baud = 9600
fileName = "D:\\ankita\\ankita_project\\final_project_code\\datasets\\emg_val_status.csv"
ser = serial.Serial(esp_port, baud)
print("Connected to ESP32 at port " + esp_port)
sample = 10000
line = 0
# Start collecting data
prompt = input("Start collecting data (y/n): ")
if prompt == 'y':
    print("Wait for 1 second....")
    time.sleep(1)
    print("Start collecting EMG data")
    # To print data with label and status
    with open(fileName, 'w', newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(["EMG Value", "Status"])  # Write header
        # Read serial port
        while line < sample:
            raw_data = ser.readline()
            data_str = raw_data.decode('utf-8').strip()
            # Split "value,status" format
            if ',' in data_str:
                value, status = data_str.split(',', 1)
                print(value + ',' + status)
```

```
            print()
            writer.writerow([value, status])
            line += 1
else:
   print("Don't have to print anything!")
```

Output:

```
484,Open
368,Open
359,Open
364,Open
357,Open
353,Open
Data Collection Completed
```

Fig. 4.2: EMG Value Collected

2. Plot dataset using matplotlib

```
import csv
import pandas as pd
import matplotlib.cbook as cbook
from matplotlib import pyplot as plt
fileName =
cbook.get_sample_data("D:\\msc_rtmnu\\major_project\\final_project_code\\datasets\\emg_train
_status.csv", asfileobj=False)
with cbook.get_sample_data(fileName) as file:
   fileName = pd.read_csv(fileName)
```

fileName.plot()

plt.show()

Output:

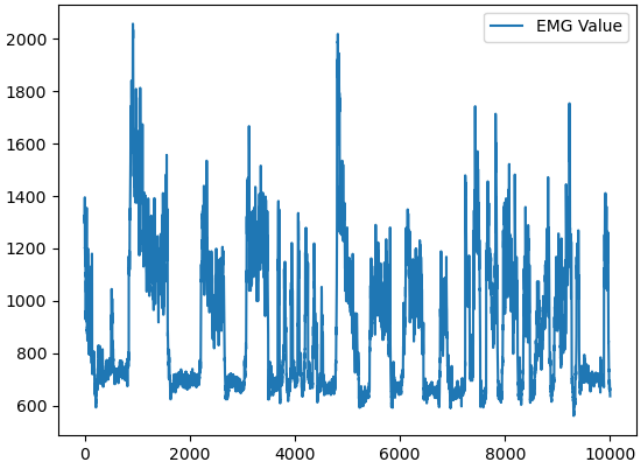


Fig. 4.3: Raw EMG Graph

3. Model Training

```
import os
os.environ["TF_ENABLE_ONEDNN_OPTS"] = "0"
import pandas as pd
import numpy as np
import h5py
import tensorflow as tf
from keras import Sequential
from tensorflow import keras
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn import metrics
import pickle
```

```
from keras.layers import Dense, Dropout

data =
pd.read_csv("D:\\msc_rtmnu\\major_project\\final_project_code\\datasets\\emg_train_status.csv"
)
X = data["EMG Value"]
y = data['Status'].map({'Open': 0, 'Close': 1})

scaler = MinMaxScaler()
X = X.to_frame()
X = scaler.fit_transform(X)

# Train test split
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)
# print("Count y_train labels")
# print(y_train.value_counts())

model = tf.keras.Sequential([
    tf.keras.layers.Dense(32, input_shape=(1,)),
    Dropout(0.25),
    # Add Batch Normalization layer
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Activation('relu'),
    Dropout(0.5),
    tf.keras.layers.Dense(1),
    tf.keras.layers.Activation('sigmoid')
])

model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
model.fit(X_train, y_train, batch_size=32, epochs=150, validation_data=(X_val, y_val))
```

```
y_pred = model.predict(X_val)
y_pred = y_pred.round()
y_pred = y_pred.reshape(len(y_pred))
print(y_pred)

# Evaluate on training data
train_loss, train_acc = model.evaluate(X_train, y_train, verbose=0)
print(f"Training Accuracy: {train_acc:.4f}")

# Evaluate on test data
loss, accuracy = model.evaluate(X_val, y_val, verbose=0)
print(f"Model Accuracy: {accuracy:.4f}")
print(f"Loss: {loss:.4f}")
print()
confusion_matrix = metrics.confusion_matrix(y_val, y_pred)
print("Confusion Matrix: ")
print(confusion_matrix)
print()
f1 = f1_score(y_val, y_pred)
print(f"F1 Score: {f1}")

# Save model to disk using pickle
with open("model_pkl", "wb") as f:
    pickle.dump(model, f)

# Save the scaler for later use
with open('scaler_pkl', 'wb') as f:
    pickle.dump(scaler, f)
```

Output:

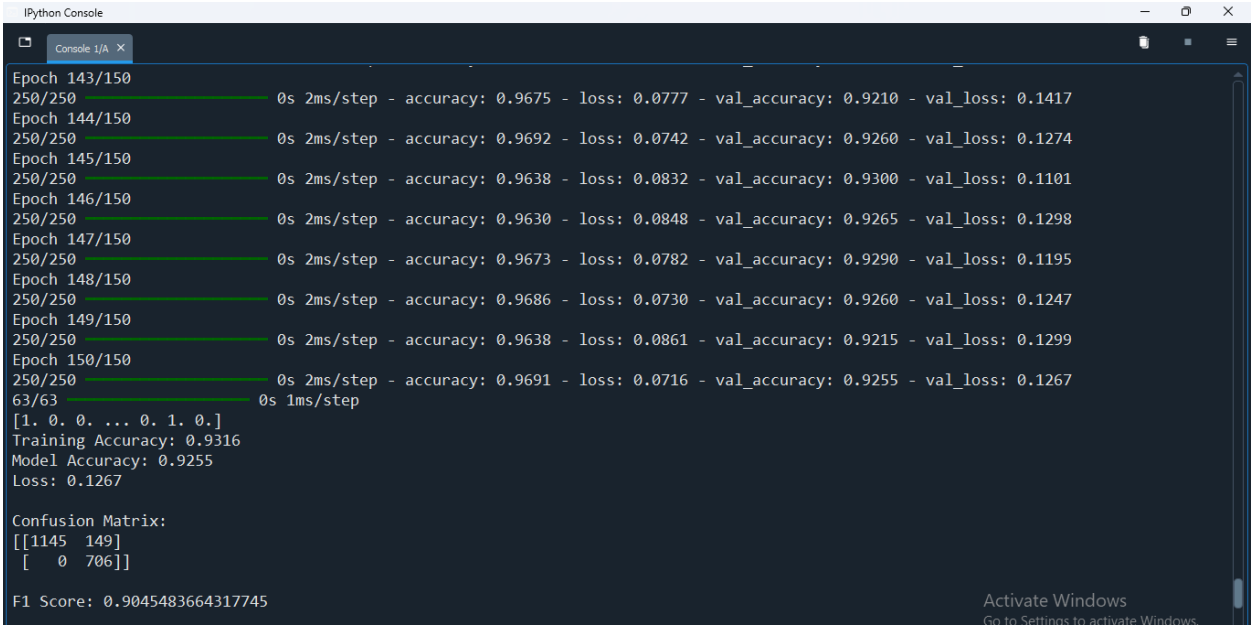


Fig. 4.4: ANN Model Accuracy

4. Testing on real time data

```
import os
os.environ["TF_ENABLE_ONEDNN_OPTS"] = "0"
import serial
import csv
import time
import numpy as np
from sklearn.preprocessing import MinMaxScaler
import pickle

# Serial configuration
esp_port = 'COM3'
baud = 9600
fileName = "D:\\ankita\\ankita_project\\final_project_code\\datasets\\predictions1.csv"
ser = serial.Serial(esp_port, baud)
```

```
print(f"Connected to ESP32 at {esp_port}")

sample_size = 2000
current_sample = 0
correct_predictions = 0

prompt = input("Start collecting data (y/n): ")
if prompt.lower() == 'y':
    print("Initializing...")
    time.sleep(1)

    with open(fileName, 'w', newline='') as csvfile:
        writer = csv.writer(csvfile)
        writer.writerow(["EMG Value", "True Status", "Predicted Status"])

        while current_sample < sample_size:
            raw_data = ser.readline()
            try:
                data_str = raw_data.decode('utf-8').strip()
                if ',' not in data_str:
                    continue

                value_str, true_status = data_str.split(',', 1)
                emg_value = float(value_str)

                # Normalization
                scaled_value = scaler.transform([[emg_value]])

                # Prediction
                prediction = model.predict(scaled_value)
                predicted_status = 'Close' if prediction.round().astype(int)[0] == 1 else 'Open'
```

```
            # Check Accuracy
            if predicted_status == true_status.strip():
                correct_predictions += 1

            # Display data on screen
            writer.writerow([emg_value, true_status, predicted_status])
            print(f"Value: {emg_value:.2f} | Prediction: {predicted_status}")
            current_sample += 1

        except (ValueError, UnicodeDecodeError) as e:
            continue

    # Final accuracy report
    accuracy = (correct_predictions / sample_size) * 100
    print(f"\nSystem Accuracy: {accuracy:.2f}%")

else:
    print("Operation cancelled.")
```

Output:

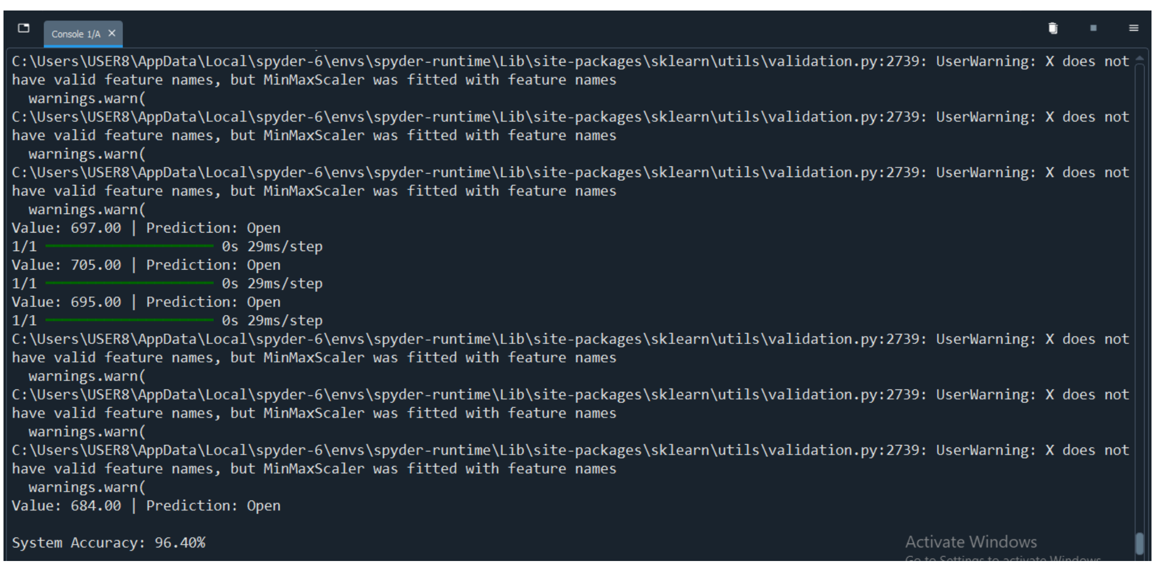


Fig. 4.5: Accuracy on real time data

4.2 Result and Discussion

The present system is designed to control servo motors mounted on an exoskeleton by reading muscle data from a healthy hand using an EMG sensor. Two ESP32 microcontrollers are used to transmit the EMG data to the servo motors via the ESP-NOW protocol, enabling wireless communication. The muscle data from the healthy hand is collected in a CSV file and is used to train an Artificial Neural Network (ANN), which classifies the data into two categories: open or closed hand gestures. The model achieves an accuracy of 83% after running 10 epochs. To improve accuracy, a more robust neural network with additional epochs and hidden layers can be developed. Once accurate results are predicted using real-time data, the model will be deployed onto the microcontroller for operational use.

4.3 Conclusion

The present system employs a traditional method to control the movement of a wearable exoskeleton by setting a threshold value for muscle activation. This system utilizes two microcontrollers: one to send the EMG signal and the other to receive the signal and operate the servo motors. The ESP-NOW protocol is used to establish a connection between the two ESP microcontrollers. This traditional method works accurately but fails at certain point when range of EMG signal changes. The muscle signal differs from person-to-person which may bring inconsistency to the system. To resolve this problem a novel technique of machine learning has been employed. The machine learning model recognizes the pattern in which muscle signal has been firing. Currently, the system is trained on neural network. Signal processing is important for training the model to recognize patterns accurately. EMG sensors produce noise that can interfere with data accuracy. We can reduce this noise by using a Butterworth bandpass filter, which removes unwanted low and high frequencies, resulting in clearer signals for the model. The system employs TowerPro SG90 servo motors to control exoskeleton, but is not much powerful to pull hand. To increase torque PCA9685 servo motor driver has been integrated with the microcontroller.

4.4 Limitation of the Present Work

The current system focuses on basic hand opening and closing actions, can be expanded to classify diverse hand gestures through machine learning integration by leveraging EMG signal processing. To achieve robust flexion and extension, high-torque servo motors like the MG996R with 9.4 kg/cm torque can be implemented, ensuring sufficient force for natural movements of the exoskeleton. The exoskeleton design can multiple degrees of freedom (DOFs) at key joints to mimic natural movement while providing a lightweight and efficient system to the patient. Integrating a microcontroller with expanded memory and machine learning enabled like ESP32 S3, enables real-time gesture classification and adaptive control algorithms, improving patient-specific customization.

4.5 Scope for Future Work

The present work can be expanded to enable individual finger movements for more precise control, which is particularly valuable in rehabilitation for patients recovering from hand injuries or strokes. Such fine-tuned finger movement could help individuals relearn delicate tasks like writing, grasping objects, or typing, improving their independence and quality of life. Additionally, wrist movements can be incorporated by installing more servos on the exoskeleton, allowing users to perform natural arm rotations. Furthermore, the exoskeleton can be adapted for use on other parts of the body, such as the legs or torso, to assist individuals with spinal cord injuries or mobility disorders. This system has further applications in various fields. In healthcare, it could allow doctors to perform remote surgeries with robotic precision or help therapists guide patients through personalized rehabilitation exercises. For industrial use, it could enhance safety by controlling robotic arms in mining operations or managing manufacturing machinery in high-risk environments. Future advancements could integrate AI to adapt assistance levels in real time, use lightweight materials for comfort, and improve energy efficiency for longer operation. By combining precise movement control with full-body adaptability, this system offers a versatile solution for improving mobility, safety, and productivity across medical, industrial, and everyday settings.

References

[1] Francesco Di Nardo, Antonio Nocera, Alessandro Cucchiarelli, Sandro Fioretti and Christian Morbidoni, "**Machine Learning for Detection of Muscular Activity from Surface EMG Signals**", Sensors, 22(3393), pp. 1-17, 2022

[2] Ankita L. Dharmik, S. K. Atre, S. J. Sharma, "**Design of an Exoskeleton for Hand Rehabilitation**", IEEE Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI), 1(1), pp. 1-4, 2022

[3] Mailyn Calderon-Diaz, Rony Silvestre Aguirre, Juan P. Vasconez, Roberto Yanez, Matias Roby, Marvin Querales, Rodrigo Salas, "**Explainable Machine Learning Techniques to Predict Muscle Injuries in Professional Soccer Players through Biomechanical Analysis**", Sensors, 24(119), pp. 1-13, 2024;

[4] Ching Yee YONG and Terence Tien Lok SIA, "**I Want to Control Your Move: Human-Human Interface (HHI) Neuromuscular Electrical Stimulator (NMES)"**, Conference of Electronics, Communication and Networks (CECNet), 1(1), pp. 724-730, 2022

[5] Nestor J. Jarque-Bou, Joaquin L. Sanche-Bau, Margarita Vergara, "**A Systematic Review of EMG Applications for the Characterization of Forearm and Hand Muscle Activity during Activities of Daily Living: Results, Challenges, and Open Issues**", Sensors, 21(3035), pp. 1-26, 2021

[6] Yassine Bouteraa, Ismail Ben Abdallah, Khalil Boukthir, "**A New Wrist-forearm Rehabilitation Protocol Integrating Human Biomechanics and SVM -based Machine Learning for Muscle Fatigue Estimation**", Bioengineering, 10(219), pp. 1-22, 2023

[7] Md. Latifur Rahman, Md. Jahin Alam, Nayeed Rashid, Lamiya Hassan Tithy, Alfaj Uddin Ahmed, and M Tarik Arafat, "**IoT Based Cost Efficient Muscle Stimulator for Biomedical Application**", 2020 IEEE Region 10 Symposium (TENSYMP), 1(1), pp. 1-5, 2020

[8] Nat Wannawas and Aldo Faisal, "**Towards AI-controlled FES-restoration of Arm Movements: Controlling for Progressive Muscular Fatigue with Gaussian State-space Models**", 11th International IEEE/EMBS Conference on Neural Engineering (NER), 1(1), pp. 1-4, 2023

[9] Dr. G. Sophia Reena, "**Posturer Guardian with Smart Muscle Strain Detection and Correction using TinyML**", Journal of Propulsion Technology, 44(4), pp. 5187-5194, 2023

[10] DERVİŞ PAŞA, "**Development of Wireless Microcontroller Based Functional Electronic Stimulation Device for Drop Foot Correction**", Near East University, Nicosia, pp. 1-87, 2014

[11] Xiaoshi Chen, Li Gong, Lirong Zheng, and Zhuo Zou, **"Soft Exoskeleton Glove for Hand Assistance Based on Human-Machine Interaction and Machine Learning",** IEEE International Conference on Human-Machine Systems (ICHMS), Rome, Italy, 1(1), 2020

[12] Xiaoshi Chen, Li Gong, Liang Wei, Shih-Ching Yeh, and Li Da Xu, **"A Wearable Hand Rehabilitation System with Soft Glove"**, IEEE Transactions on Industrial Information, 17(2), pp. 943-952, 2020

[13] Alireza Mohammadi, Jim Lavranos, Peter Choong, and Denny Oetomo, **"Flexo-glove: A 3D Printed Soft Exoskeleton Robotic Glove for Impaired Hand Rehabilitation and Assistance"**, 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Honolulu, USA, pp. 2120-2123, 2018

[14] Waqas Ahmed, Muhammad Kashif Sattar, Wajeeha Shahnawaz, Umair Saeed, Shahbaz Mehmood Khan, and Neha Amon Khan, **"Wearable Hand-Rehabilitation System with Soft Gloves for Patient with Face Paralysis and Disability"**, Engineering Proceedings, 12(56), pp. 1-5, 2021

[15] Xuanyi Zhou, Hoa Fu, Baoqing Shentu, Weidong Wang, Shibo Cai, and Guanjan Bao, **"Design and Control of a Tendon-Driven Robotic Finger Based on Grasping Task Analysis"**, Biomimetics, 9(370), pp. 1-17, 2024

Annexure

Ministry of Education
Government of India

Pradhan Mantri Uchchatar Shiksha Abhiyan
Ministry of Education, Government of India

Sponsored & In Association with

POST GRADUATE TEACHING DEPARTMENT OF LAW
PROTECTION CELL AGAINST SEXUAL HARASSMENT & GENDER DISCRIMINATION AT WORKPLACE
Internal Quality Assurance Cell (IQAC)
Rashtrasant Tukadoji Maharaj Nagpur University

Three Days Workshop on

"Cultural Diversity and Sensitivity"

29-31 January 2025

Certificate

This is to certify that

Dr./Mr./Mrs./Miss Ankita S. Girhoti has participated/presented as paper titled A Survey on Wearable Exoskeleton for a paralysed Hand in the National Conference on **"FOSTERING CULTURAL DIVERSITY AND SENSITIVITY IN HIGHER EDUCATIONAL INSTITUTION: PERSPECTIVE AND CHALLENGES IN INDIA"** organised on 29th January 2025 and has secured ..... position in paper presentation.

Dr. Smita Acharya
Convener
Director, IQAC
RTM Nagpur University

Dr. Shalini Lihitkar
Co-ordinator
Prof, Dept. of Library & Info. Sci.
RTM Nagpur University

Dr. Payal Thaorey
Co-ordinator
Head, PGTD Law
RTM Nagpur University