MyoWave

Transforming electrical signals into motion!

written

by

Kabita Bhusal (Matriculation no.: 8119930), kabita.bhusal@sport.uni-giessen.de

for

Computer Programming in Human Movement Analytics - Final Project
(MA-HMA-05, MA-BMB-04)
as part of course
Msc in Human Movement Analytics: Biomechanics, Motorcontrol & Learning (WiSe 2024-25)

Faculty 06, Department of Sports Science
Lecturers: M. Sc Lea Junge-Bornholt

Date of Submission: 16-03-20225

Contents

List of Figures

1 Introduction

Hand gesture classification is a crucial area of research in human-computer interaction(HCI), particularly for applications in prosthetics, robotics, and assistive technologies. By integrating Electromyography (EMG) data, gesture recognition systems can achieve higher accuracy, as EMG signals provide detailed movement-related information. However, these signals are often noisy and require effective preprocessing techniques such as filtering. Additionally, selecting appropriate features and machine learning models is essential for achieving optimal classification performance.

To simplify this complex process, **MyoWave**, a MATLAB-based application, has been developed. MyoWave provides an intuitive platform that enables users to preprocess EMG data, train machine learning models, and evaluate classification performance without requiring extensive coding knowledge. Using artificial intelligence techniques such as Machine Learning (ML) and Deep Learning (DL), Myowave aims to enhance accessibility and efficiency in hand gesture classification, making it a valuable tool for researchers and developers in this field.

1.1 Objectives

The main objectives of MyoWave are:

- Develop a simple framework for training and testing machine learning models for EMG-based hand gesture classification.
- Implement signal preprocessing techniques to remove noise and improve data quality.
- To provide a user-friendly MATLAB GUI for seamless interaction and visualization of results.

1.2 Scope

MyoWave focuses on offline EMG signal classification for a predefined set of hand gestures. The system will allow users to upload EMG data, apply preprocessing filters, train machine learning models, and visualize classification results. While the current version is limited to machine learning-based classification, future improvements could include real-time processing, deep learning integration, and compatibility with

wearable EMG devices for broader applications in HCI, biomedical engineering, and rehabilitation.

2 Methodology

The methodology of this project begins with the selection of an online dataset from ScienceDirect, titled *"Dataset for Multi-Channel Surface Electromyography (sEMG) Signals of Hand Gestures"*(https://www.sciencedirect.com). This dataset comprises four-channel electromyography (EMG) data recorded from 40 subjects performing 10 different hand gestures. The four channels were strategically positioned on the extensor carpi radialis, flexor carpi radialis, extensor carpi ulnaris, and flexor carpi ulnaris muscles to ensure accurate and comprehensive signal acquisition.

The dataset was recorded at a high sampling rate of 2 kHz, ensuring high-quality signal capture. Each participant performed the hand gestures in five repetitions, but due to time constraints, only data from one repetition was utilized in this study. The experimental protocol involved performing each hand gesture for six seconds, with a rest period of four seconds before and after each gesture to minimize muscle fatigue and baseline noise.

Figure 1 illustrates the various hand gesture positions involved in the dataset, while Figure 2 provides a visual representation of the experimental recording timeline.

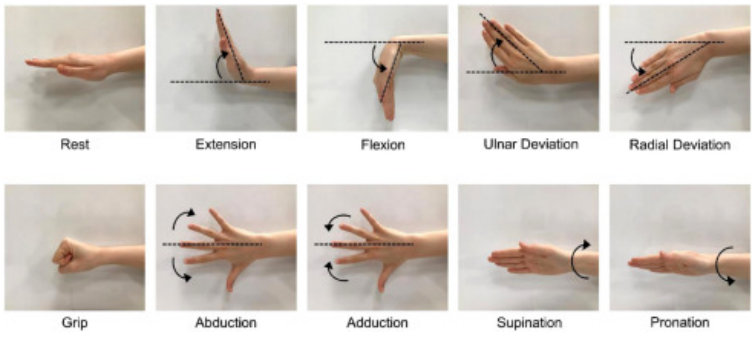


Figure 1: Hand gesture positions

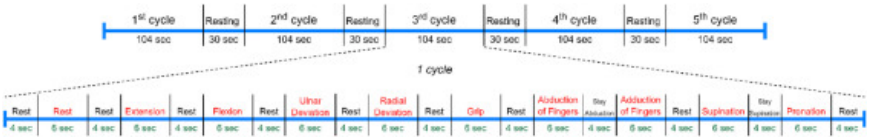


Figure 2: time line of recording

2.1 Preprocessing

Before initiating the preprocessing stage, data segmentation was performed to separate datasets corresponding to different hand positions. This was essential for effective machine learning model training, ensuring that each hand position was distinctly categorized. Due to time constraints and complexity, the segmentation code was not included in the GUI project. Since the time duration and sequence of hand positions, including rest periods, were already known, a "for loop" was used to segment the data systematically for all subjects by specifying the indices for each hand position and the number of corresponding data points. The segmented data was then saved using subject numbers and position names to ensure structured storage and accessibility for further processing.

The preprocessing of EMG data commenced with importing the raw data into the MATLAB GUI. The "uigetfile" function was employed to enable users to select the file and path, while the "fullfile" function was used to construct the complete file specification, ensuring seamless file handling. The raw EMG data was stored in the variable *'rawData'* for subsequent processing.

To eliminate noise and motion artifacts, two distinct filtering techniques were employed: the Moving Average (Movmean) filter and the Butterworth filter. These techniques were applied separately on the basis of the characteristics and requirements of the data.

The Moving Average filter was implemented using the "movmean" function with a window size of 100. Considering the slow movement nature of the EMG data and a sampling frequency of 2000 Hz, each data point represented 0.5 ms. A 50 ms window corresponded to 100 data points, making it suitable for noise reduction while preserving essential signal characteristics.

A 4th order Butterworth filter with a cutoff frequency (Fc) of 50 Hz was applied using the "filtfilt" function to ensure zero-phase distortion. The choice of a lower cutoff frequency was made to prevent excessive smoothing and preserve signal

integrity. Various window sizes, cutoff frequencies, and filter orders were tested to determine optimal parameters, ensuring that the structure of the EMG signal remained unchanged.

It should be noted that, due to advancements in amplification technology, filtering may not always be necessary for EMG data, depending on signal quality. All processed data was stored in the structure array *'dataObject.processedData'* for further analysis. Figures 3 and 4 present the raw and filtered data of a single subject and hand position, illustrating the effects of Butterworth and Movmean filtering techniques.

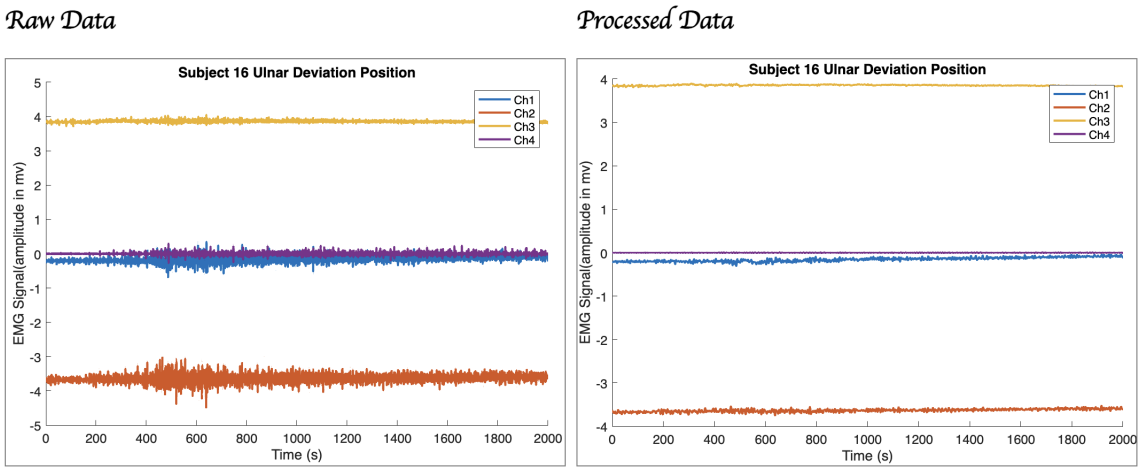


Figure 3: Raw data and filtered data(butterworth)

Raw Data *Processed Data*

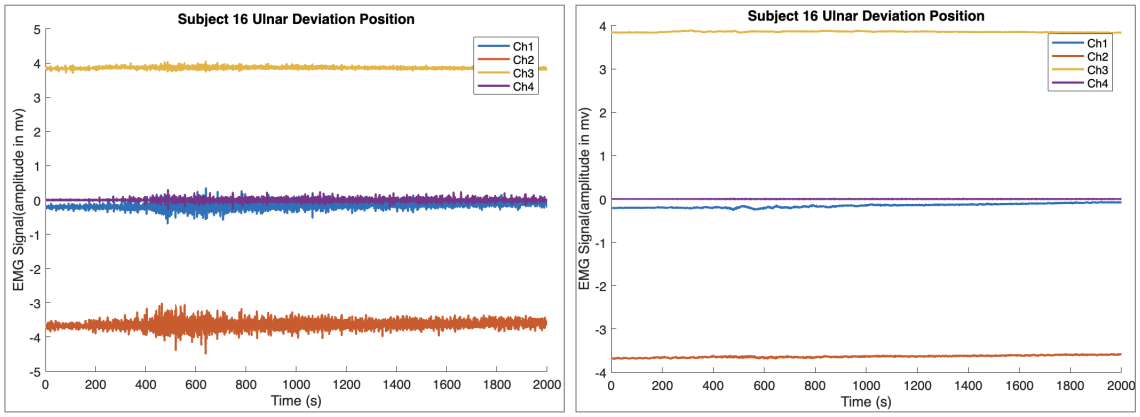


Figure 4: Raw data and filtered data(movmean)

2.2 Machine Learning Model

The MyoWave application utilizes a Decision Tree classifier to classify hand gestures using electromyography (EMG) signals. The data set consists of 40 subjects, each performing hand gestures in multiple positions, recorded on 4 EMG channels.

2.2.1 Data Preparation and Reshaping

original Data Structure: Each subject's EMG signal was originally stored in a matrix format where

- Each subject's data contains 4 EMG channels.
- Each channel records 12,000 data points per hand position.

This results in a 12000 $\times$ 4 matrix for each subject per position.

Data Reshaping for Model Training: To prepare the dataset for machine learning, the data was reshaped to ensure that all four channels of each subject's position were stored in a single row. The transformation resulted in:

- Training Dataset(75%) : 300 x 48000
- Testing Dataset(25%) : 100 x 48000

Each row in the final dataset represents the combined EMG signals from four channels for a single hand position of a subject, resulting in a 48,000-feature vector per sample.

2.2.2 Dataset Splitting and Storage

The dataset was split into training (75%) and testing (25%) sets, stored in MATLAB as follows:

- Training Data:
  - EMG feature data stored in *dataObject.modelPackage.trainX*
  - Corresponding labels stored in *dataObject.modelPackage.trainY*
- Testing Data:
  - EMG feature data stored in *dataObject.modelPackage.valX*
  - Corresponding labels stored in *dataObject.modelPackage.valY*

The structured storage allows for efficient data access and processing during training and evaluation.

2.2.3 Decision Tree Model Training

A Decision Tree classifier ("fitctree") was chosen due to:

- Its ability to process high-dimensional EMG data efficiently.
- Its suitability for datasets with structured numerical patterns.
- Its interpretability and ease of implementation.

Model Training Process: The classifier was trained using 300 training samples (trainX), with corresponding labels (trainY). The trained model was then stored in *dataObject.modelPackage.model* for future testing and classification.

2.2.4 Model Testing and Evaluation

The trained model was evaluated using 100 test samples (valX), and the predictions were compared with the actual labels (valY).

The total number of correct predictions was summed, and the overall accuracy percentage was calculated by dividing this value by the total number of testing positions. Figure 5 presents the overall accuracy, performance metrics of the model, and a test example for a single position. However, the accuracy may vary with different filtering techniques and each instance of model training.

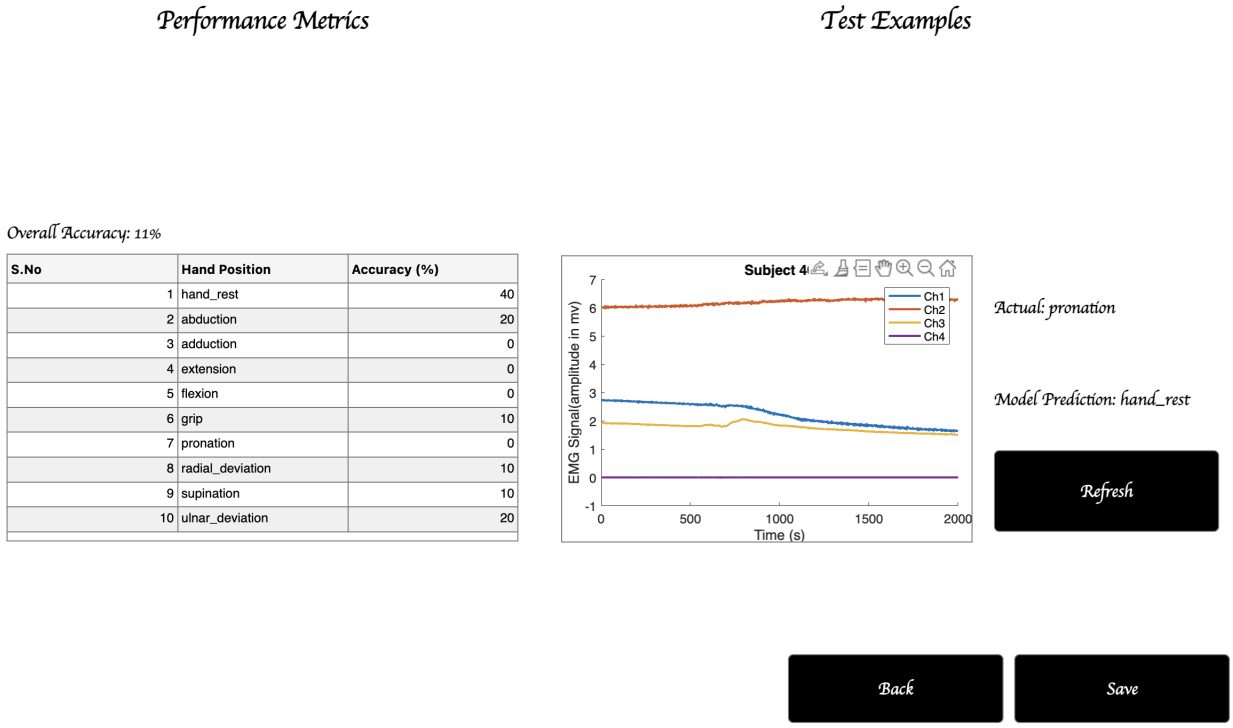


Figure 5: Performance metrix

3 Implementation of MyoWave

Important Note: Run only the MyoWave.mlapp file while ensuring all other files remain in the same folder to open the application.

The MyoWave application features an interactive Graphical User Interface (GUI)

designed to streamline the process of hand gesture classification using EMG signals. The GUI enables users to load, preprocess, train, and test their data efficiently while providing clear visual feedback through graphs, buttons, and notifications.

The interface is structured into multiple pages, guiding users through each step in a sequential manner, ensuring an intuitive and error-free workflow.

3.1 Main Window – File Loading

Upon launching MyoWave, users are greeted with the Loading Page, displaying:

- App Name & Tagline
- A **"Load"** button for selecting a dataset.

Note: Please select the *"processed.zip"* file provided in the folder to proceed to the next page of the application. Ensure that only *.zip* file is selected, as the application is designed to accept this format exclusively.

Once a valid file is selected, the application proceeds to the Main Dashboard. Figure 6 illustrates the appearance of the main window – loading page.

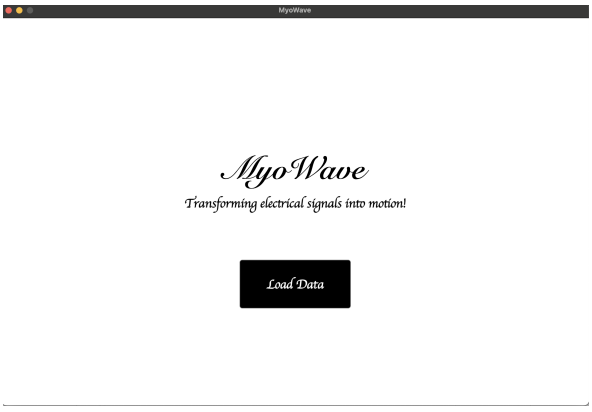


Figure 6: Main window

3.2 Main Dashboard(Main Page) – Navigation to Key Functions

The Main Page consists of three core buttons:

- Data Preprocessing (Enabled)
- Train Model (Disabled initially)
- Test Model (Disabled initially)

The **"Train Model"** and **"Test Model"** buttons remain disabled until data preprocessing is complete.

Figure 7 presents the visual layout of the main page.

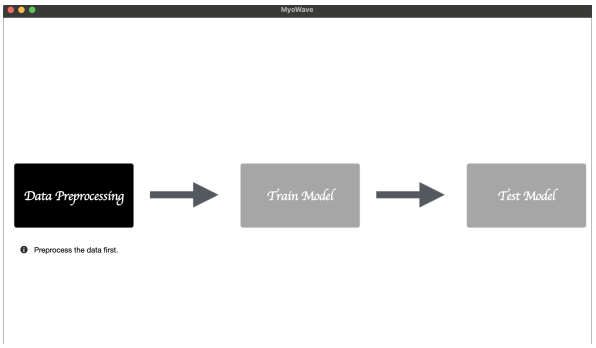


Figure 7: Main page

3.3 Data Preprocessing Page

After clicking the **"Data Preprocessing"** button on the Main Page, users are directed to the Preprocessing Page, where the raw EMG data is initially displayed.

3.3.1 Viewing Raw EMG Data

- Users can see the raw EMG signal graph as soon as they enter the preprocessing page.
- **Dropdown menus** allow the selection of different ***subjects*** and ***hand positions*** to visualize the corresponding raw EMG data.

Figure 8 represents the raw EMG signal graph as displayed in the application, showing the unprocessed data before applying any filtering techniques.

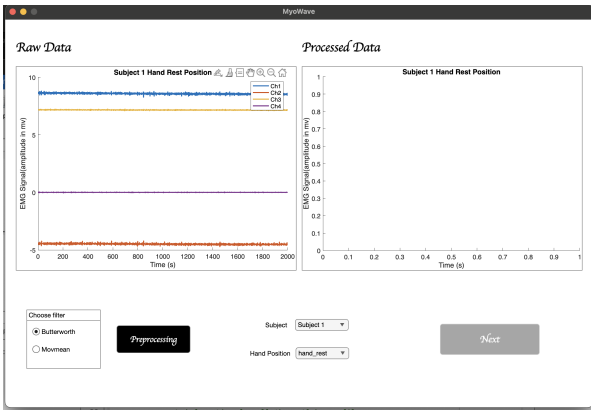


Figure 8: Preprocessing page(raw EMG graph)

3.3.2 Selecting a Filtering Technique

A **radio button** group named ***"Choose filter"*** offers two filtering options for preprocessing the EMG data.

- Movmean Filter
- Butterworth Filter

Users can select their preferred filter and press the **"Preprocess"** button to apply it, as shown in Figure 8.

3.3.3 Comparing Raw and Processed Data

- Once filtering is applied, a processed data graph appears alongside the raw data.
- Users can switch between different subjects and positions using dropdowns to compare the effect of preprocessing.

Figure 9 shows the appearance of the Preprocessing Page after applying the selected filter.

Once preprocessing is completed, the **"Next"** button is enabled, allowing users to proceed, which was previously disabled, as shown in Figure 8.

Figure 9: Preprocessing page(raw and filtered EMG graph)

3.4 Model Training Page

After preprocessing, the **"Train Model"** button on the Main Page becomes enabled. Clicking on it opens the Training Page, which includes:

- A "Train Model" button to start training.
- A "Next" button, initially disabled.

Figure 10 presents the visual layout of the training page.

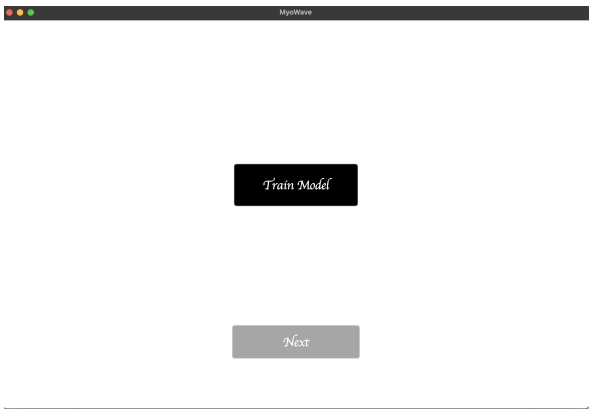


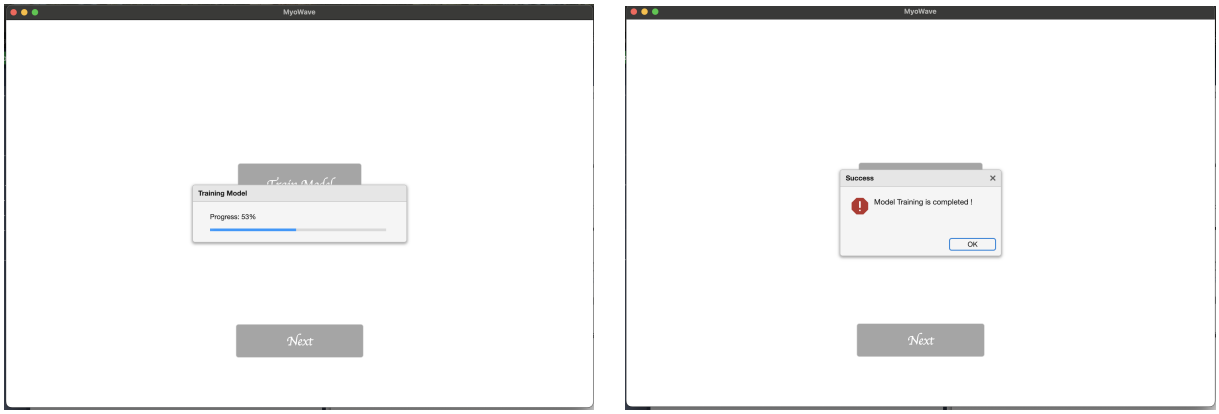
Figure 10: Train model page

3.4.1 Model Training Process

- Clicking "Train Model" starts training, accompanied by a progress bar.

- Upon completion, a notification appears, confirming "Model training is completed". Press OK to continue.
- The "Next" button becomes enabled, allowing users to proceed.

Figure 11 shows the appearance of the progress bar and completion notification during the model training process.



(a) (b)

Figure 11: Training page (a) Progression bar, (b) Notification

3.5 Testing the Model and Performance Evaluation

After training, the **"Test Model"** button on the Main Page becomes enabled. Clicking on it opens the Testing Page, which displays:

- Overall model accuracy
- A table showing hand positions and accuracy percentages.
- Graph hand position with labels, where the actual hand position and the prediction of the model are written.
- Users can click the **"Refresh"** button to view predictions for different test subjects. Since only 25% of subjects are used for testing, the refresh button cycles through last 10 cases.

3.5.1 Saving the Model and Results

At the bottom of the Testing Page, two options are available:

- **"Save"** button – Saves the trained model and accuracy table to the GUI's folder.
- **"Back"** Button – Returns users to the main page for further refinement.

Figure 5 above shows the overall appearance of the Testing Model Page.

3.6 Workflow Flexibility and Iterative Improvements

The MyoWave GUI allows users to refine their workflow:

- Users can return to preprocessing and apply a different filter.
- They can train the model again without preprocessing.
- After training, they can test the model and save results.

4 Result and Discussion

The Decision Tree model implemented in MyoWave showed low accuracy, achieving 15% with the Movmean filter and 11% with the Butterworth filter. While Movmean filtering provided smoother data, the overall performance remained poor, likely due to the unsuitability of the Decision Tree model for time-series EMG data. More advanced models, such as Convolutional Neural Networks (CNNs), could improve accuracy.

Despite this, the GUI of MyoWave is highly user-friendly, offering:

- Easy navigation through preprocessing, training, and testing steps.
- Raw and processed EMG data visualization with filtering options.
- Progress tracking during model training.
- Testing results display, including accuracy metrics and model predictions.

While accuracy is a limitation, the application provides a well-structured framework for EMG-based hand gesture classification, with potential for further improvements in model selection and filtering techniques.

5 Challenges and Difficulties

Low Model Accuracy: The Decision Tree model used for classification yielded low accuracy. The main difficulty was selecting an appropriate machine learning model for time-series EMG data, which may require more advanced approaches.

Handling Large Data Size: Each subject's EMG data consisted of four channels with 12,000 data points, requiring reshaping into large matrices (300×48,000 for training and 100×48,000 for testing).This process was computationally intensive and required efficient data handling techniques to ensure smooth processing and analysis.

Preprocessing Complexity: Implementing filtering techniques required careful design to ensure proper visualization of raw and processed data while maintaining usability. Managing dynamic updates in the GUI when users changed subjects or positions added to the complexity.

GUI Navigation and Interactivity: Ensuring a smooth workflow was challenging, particularly with enabling/disabling buttons based on the process stage (e.g., enabling "Next" only after preprocessing).

Testing and Performance Evaluation: Since only 25% of the dataset was used for testing, performance metrics had to be displayed effectively. Additionally, incorporating a graph and label of actual hand positions versus model predictions required careful formatting for clear visualization.

6 Future Improvements

Several enhancements can be made to improve the MyoWave application:

Real-time EMG Signal Processing: Enable real-time EMG signal acquisition for immediate classification. Instead of using pre-segmented data, the application can be modified to acquire and process EMG signals in real time within the GUI.

Improved Machine Learning Models: Since Decision Trees are not well-suited for time-series data, future versions can incorporate models like Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), or Hidden Markov

Models (HMMs) to enhance accuracy. Deep learning approaches could also be explored for better feature extraction and classification.

Comparative Model Training: Users could be given the option to train the model with and without preprocessing, allowing a direct comparison of the impact of filtering techniques on classification accuracy.

Expansion of Gesture Classes: Additional hand gestures can be included to improve the versatility and real-world applicability of the system.

These improvements will enhance both accuracy and usability, making MyoWave more effective for EMG-based hand gesture recognition.