EMG Signal Processing for Prosthetic Control

Anirudh - CB.EN.U4CCE22006

Anna R Riya Abbhishree- CB.EN.U4CCE22013

Kaavyaa Aravind - CB.EN.U4CCE22057

Department of Electronics and Communication Engineering,

Amrita School of Engineering, Coimbatore,
Amrita Vishwa Vidyapeetham, India

***Abstract*— This research endeavors to simulate the control of a virtual hand through the analysis of electromyography (EMG) signals employing Digital Signal Processing (DSP) techniques. The comprehensive procedure encompasses distinct phases, including feature extraction, classification, and signal preparation. Raw EMG signals obtained from a text file undergo a series of processing procedures, such as butterworth bandpass filtering, rectification, and a moving average system, to enhance signal quality. Additional frequency domain information is extracted using the modified Haar wavelet technique, thereby augmenting the utility of surface electromyography (sEMG) data, with a specific focus on hand movements. Utilizing simulators like MATLAB, the project designs the output for projection. Visual representations of original sEMG signals and their wavelet transforms facilitate a better understanding of the effects of the processing stages. Post-processing techniques, including a 12-point moving average filter and other smoothing methods, contribute to a clearer signal. The entire dataset undergoes the Haar wavelet transform, amalgamating features from both frequency and temporal domains. The standardized features serve as training data for a K-Nearest Neighbors (KNN) classifier, enabling hand movement prediction. Evaluation metrics, including accuracy and a comprehensive classification report, provide a thorough analysis of the classifier's performance on a test set. The inclusion of a butterworth bandpass filter further refines signal processing, emphasizing the robustness and precision of the proposed methodology.**

Keywords— K-Nearest Neighbours (KNN); sEMG; Haar wavelet transform; feature extraction; moving average system; movement prediction; MAV; Butterworth filter.

I. INTRODUCTION

In the realm of prosthetic control, the integration of advanced signal processing techniques plays a pivotal role in enhancing the functionality and responsiveness of prosthetic devices, particularly those designed for hand control. Electromyography (EMG) and surface electromyography (sEMG) have emerged as indispensable tools for capturing the electrical activity generated by skeletal muscles, offering a direct and intuitive interface for prosthetic control. This paper explores the synergistic application of EMG and sEMG signals, coupled with the Haar wavelet transform and the k-nearest neighbors (KNN) algorithm, to address the challenges associated with effective hand prosthetic control.

EMG and sEMG signals, derived from the electrical impulses produced during muscle contraction, provide a direct reflection of the user's intended movements. Leveraging these signals for prosthetic control requires sophisticated signal processing techniques to extract meaningful features and improve the accuracy of gesture recognition. One such technique employed in this study is the Haar wavelet transform, known for its ability to capture both time and frequency-domain information. The Haar wavelet transform allows for efficient signal decomposition, providing a multi-resolution analysis that enhances the discrimination of distinct muscle activation patterns associated with different hand movements.

In conjunction with the Haar wavelet transform, the k-nearest neighbors (KNN) algorithm is implemented for classification purposes. KNN is a non-parametric, instance-based learning algorithm that assigns labels to new data points based on the majority class of their k-nearest neighbors. By employing KNN, the proposed prosthetic control system aims to robustly classify EMG and sEMG patterns, enabling real-time and accurate translation of user intentions into corresponding prosthetic actions.

This research strives to contribute to the evolving field of prosthetic control by amalgamating the inherent advantages of EMG and sEMG signals with the analytical power of the Haar wavelet transform and the classification prowess of the KNN algorithm. The integration of these techniques holds the promise of enhancing the precision and efficiency of hand prosthetic control, ultimately improving the quality of life for individuals relying on prosthetic devices for daily activities.

II. RELATED WORK

In the domain of EMG signal processing for prosthetics, effective preprocessing is crucial to enhance the accuracy of

subsequent analysis. The initial step involves capturing the electrical signals from human muscles, typically falling within the range of 5 to 250 Hz. To eliminate extraneous noise, a 4th-order Butterworth bandpass filter is commonly employed, adept at isolating muscle signals while attenuating high-frequency environmental noise and low-frequency interference like skin friction-generated noise. the transfer function of a Butterworth bandpass filter can be represented as equation (1).

$$H(z) = \frac{K \cdot (z-z_1)(z-z_2)\ldots(z-z_{2N})}{(z-p_1)(z-p_2)\ldots(z-p_N)} \quad (1)$$

The angular frequency $\omega 0$ of this circle is related to the cutoff frequencies $\omega c1$ and $\omega c2$ of the bandpass filter by equation (2)

$$\omega_0 = \sqrt{\omega_{c1} \cdot \omega_{c2}} \quad (2)$$

Despite this, there may be a need for further signal enhancement, prompting the utilization of techniques such as wavelet transform.

Wavelet transform [8,9] proves advantageous in elucidating the intricate motion dynamics of the human body by providing frequency domain information at both high and low frequencies. The Haar wavelet functions are defined as equation (3).

$$\psi(t) = \begin{cases} 1 & \text{if } 0 \le t < \frac{1}{2} \\ -1 & \text{if } \frac{1}{2} \le t < 1 \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

The Haar scaling function is defined as equation(4).

$$\phi(t) = \begin{cases} 1 & \text{if } 0 \le t < 1 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

The approximation coefficients A are detail coefficients D are given by equation (5) and (6) respectively.

$$A[k] = \frac{x[2k]+x[2k+1]}{\sqrt{2}} \quad (5)$$

$$D[k] = \frac{x[2k]-x[2k+1]}{\sqrt{2}} \quad (6)$$

The inverse Haar wavelet is given by equation(7).

$$x[n] = A[0] \cdot \phi(2n) + \sum_{k=0}^{N-1} D[k] \cdot \psi(2n-k) \quad (7)$$

Feature extraction, a pivotal phase in EMG signal processing, employs diverse methods, including autoregressive models, signal entropy measurements, and statistical assessments in the time and frequency domains. Notably, the Wavelet transform emerges as a favored approach, offering comprehensive insights across different frequency ranges.

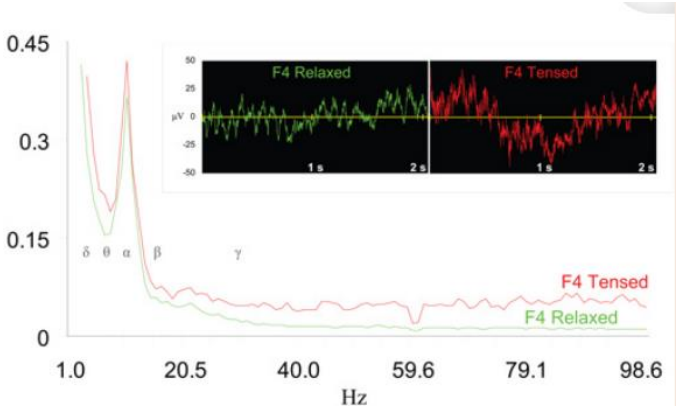
Moving window smoothing, accomplished through a moving average technique, is then applied to the signal. The moving average calculation can be efficiently calculated by applying convolution operation with an equivalent filter equal to MAV which is given by equation (8).

$$\mathrm{MAV}[n] = \frac{1}{N} \sum_{k=0}^{N-1} x[n-k] \quad (8)$$

This method involves systematically sliding a window along the time series, calculating the average within the window at each position to produce a smoothed signal. Careful consideration of the window length is paramount, balancing between preserving details and achieving smoothness. Post-smoothing, various feature extraction methods, such as time domain, frequency domain, and time-frequency analysis, are employed to extract meaningful muscle signal characteristics.

In the realm of machine learning (ML) for EMG signal classification, three prominent models—Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Ensemble classifiers—are commonly employed. Notably, the KNN model is recognized for its simplicity and effectiveness. It involves storing labeled training data and, for classifying an unlabeled object, computes a similarity metric to identify the k closest elements. The classification is then determined either by the label of the closest element (for k=1) or by a voting mechanism (for k>2).[10] The application of KNN in decomposing EMG signals into motor unit potential trains (MUPTs) showcases its utility in discerning different feature types for improved signal analysis in prosthetic applications.

Artifacts in EMG:[4]



Artifacts refer to unwanted signals or interference that can distort the accurate recording and interpretation of muscle electrical activity. These artifacts may arise from various sources, including external electrical noise, movement artifacts, and poor electrode contact. External noise, such as electromagnetic interference from nearby electronic devices, can introduce false signals into the EMG recording. Movement artifacts occur when the subject's body movements introduce additional electrical activity unrelated to muscle contractions. Additionally, inadequate electrode contact with the skin can lead to poor signal quality. Managing and minimizing artifacts is crucial in EMG studies to ensure the reliability and precision of the recorded muscle activity, allowing for accurate analysis and interpretation of neuromuscular function.

Passband Butterworth filter and Moving Average (MAV) are commonly used techniques in electromyography (EMG) signal processing to help rectify artifacts and enhance the accuracy of muscle activity recordings. A Passband Butterworth filter is employed to selectively allow certain

frequencies of the EMG signal to pass through, while attenuating others. By setting an appropriate passband, high-frequency noise and interference can be filtered out, leaving behind the relevant muscle activity signals. This helps in reducing the impact of external noise and artifacts on the EMG recordings, resulting in a cleaner and more accurate representation of muscle contractions.

Moving Average (MAV), on the other hand, is a temporal filtering technique. It involves calculating the average amplitude of the EMG signal over a specific window of time. MAV is particularly useful for smoothing out rapid fluctuations and removing short-term artifacts caused by movements or other transient disturbances. It provides a more stable representation of the muscle activity, making it easier to analyze and interpret the underlying neuromuscular patterns.

III. Proposed Methodolody

The study embraced a multifaceted exploration of EMG signal processing for prosthetics, leveraging a dataset sourced from the UCI Machine Learning Repository. Raw EMG signals, encapsulating the intricacies of specific gestures and annotated with respective class labels, underwent meticulous pre-processing. The initial step involved the utilization of a Butterworth bandpass filter, strategically implemented to mitigate noise and focus on the relevant frequency range. This filtering process, executed with a lowcut of 20 Hz and a high cut of 500 Hz, was instrumental in enhancing the signal quality, providing a foundation for subsequent analyses.

Following the bandpass filtering, the raw EMG signals were organized into a Pandas Data Frame for systematic handling. The intrinsic characteristics of these signals were elucidated through visualization, offering insights into their temporal patterns and amplitude variations across different classes.

To delve into the frequency domain information, the study employed a modified Haar wavelet transform. In addition to this transformation, the script also incorporated a Moving Average (MAV) technique. The MAV, a straightforward yet effective method, was applied to smoothen the signals and extract trends, thereby contributing valuable information for subsequent feature extraction and classification.

The impact of the bandpass filter, Haar wavelet transform, and MAV on signal representation was meticulously elucidated through visualizations. These visualizations not only provided a qualitative understanding of the signal alterations induced by each processing step but also served as a foundation for informed decision-making in subsequent stages.

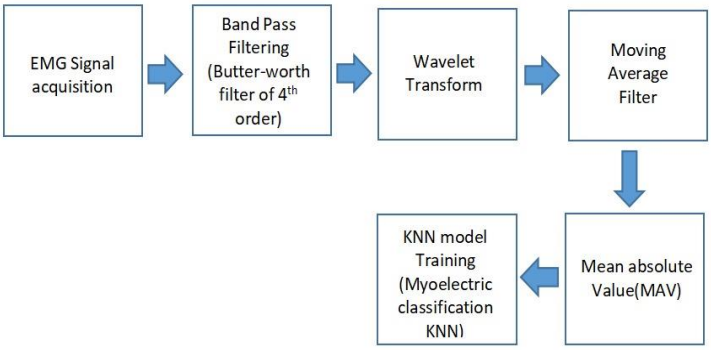
Extending these signal processing techniques to the entire dataset enabled the extraction of vital frequency domain features. Careful consideration was given to removing empty arrays resulting from problematic signals, ensuring data integrity and reliability for downstream analyses.

The fusion of time-domain features extracted through MAV and frequency domain features from the Haar wavelet transform and bandpass filtering collectively contributed to the creation of a comprehensive feature matrix (X). Corresponding class labels (y) were extracted, setting the stage for further analysis.

The dataset was judiciously split into training and testing sets, adhering to an 80-20 ratio to maintain an optimal balance for model training and evaluation. Recognizing the importance of standardization, the features underwent scaling using the StandardScaler to ensure uniformity and facilitate the effectiveness of machine learning algorithms.

In the realm of classification, the study opted for a K-Nearest Neighbors (KNN) classifier, acknowledging its simplicity and effectiveness. The classifier, configured with three neighbors (subject to experimentation-based adjustments), underwent training on the standardized training data.

The trained model, equipped with the insights gained from the bandpass filter, Haar wavelet transform, and MAV, subsequently executed predictions on the test set. The evaluation phase, encompassing accuracy calculations and the generation of a detailed classification report, offered a comprehensive assessment of the model's prowess in classifying EMG signals.



Overall, the study's amalgamation of Butterworth bandpass filtering, Haar wavelet transform, and MAV techniques in tandem with visualization, feature extraction, and machine learning presents a holistic approach to understanding and harnessing EMG signals for prosthetic applications. The utilization of a diverse set of signal processing tools underscores the meticulousness applied to enhance signal quality and extract meaningful features, contributing to the broader discourse on advancing EMG signal processing methodologies for improved prosthetic control and usability.

IV. Results and Discussions

Dataset Description: The datasets are in the form of .txt files of raw EMG data recorded by MYO Thalamic bracelet. For recording patterns, the dataset is obtained from a MYO Thalamic bracelet worn on a user's forearm, and a PC with a Bluetooth receiver. The bracelet is equipped with eight sensors

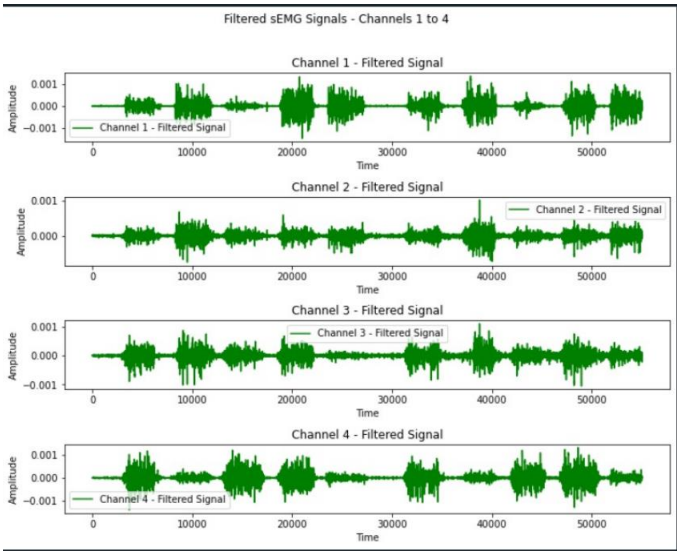
equally spaced around the forearm that simultaneously acquire myographic signals. The signals are sent through a Bluetooth interface to a PC. The datasets present raw EMG data for 36 subjects while they performed series of static hand gestures. The subject performs two series, each of which consists of six basic gestures namely resting hand, grasped hand, wrist flexion, wrist extension, ulnar deviation, radial deviation as class respectively. Each gesture was performed for 3 seconds with a pause of 3 seconds between gestures. Number of Instances is about 40000-50000 recordings in each column (30000 listed as guaranteed).

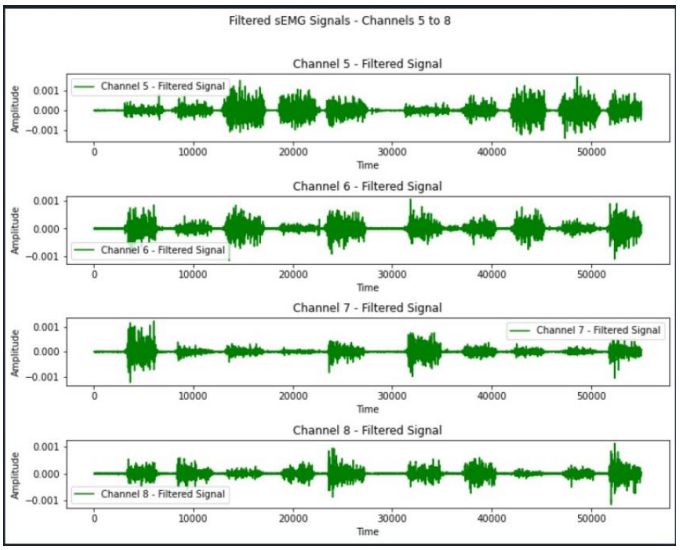
Performance metrics:

Accuracy: 0.98

Class	Precision	Recall	F1-score	Support
0	0.98	0.98	0.98	7252
1	0.96	0.95	0.95	659
2	0.98	0.97	0.98	660
3	0.96	0.97	0.96	567
4	0.95	0.95	0.95	595
5	0.97	0.96	0.97	651
6	0.96	0.96	0.96	635

	Precision	Recall	F1-score	Support
Accuracy			0.98	11019
Macro Avg	0.97	0.96	0.96	11019
Weighted Avg	0.98	0.98	0.98	11019

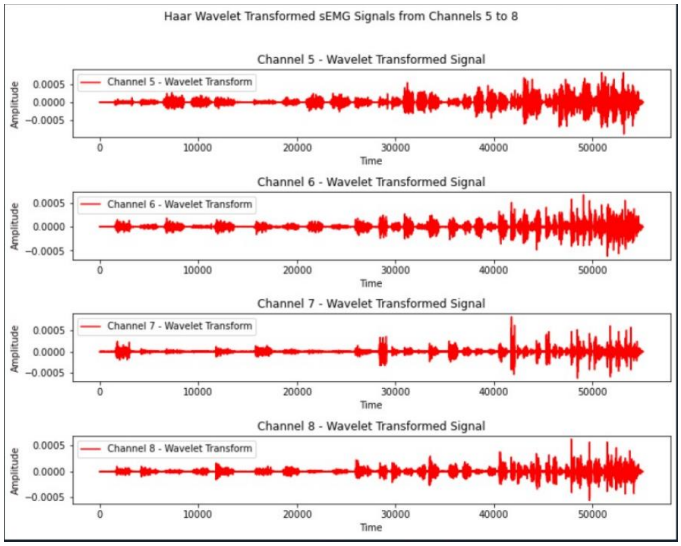


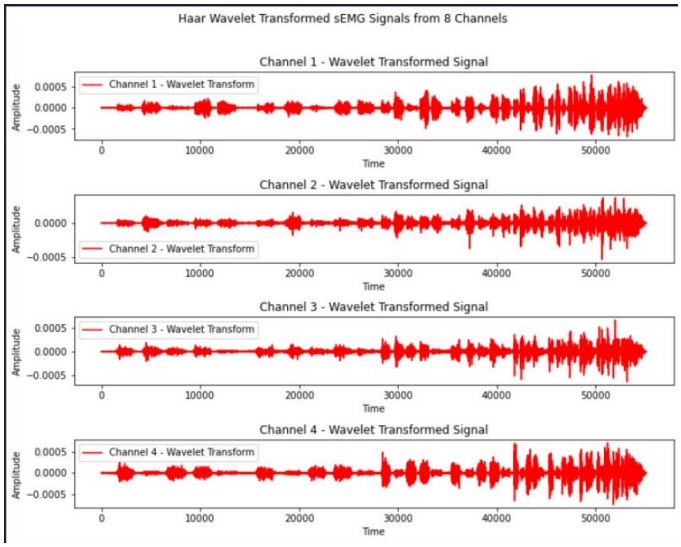


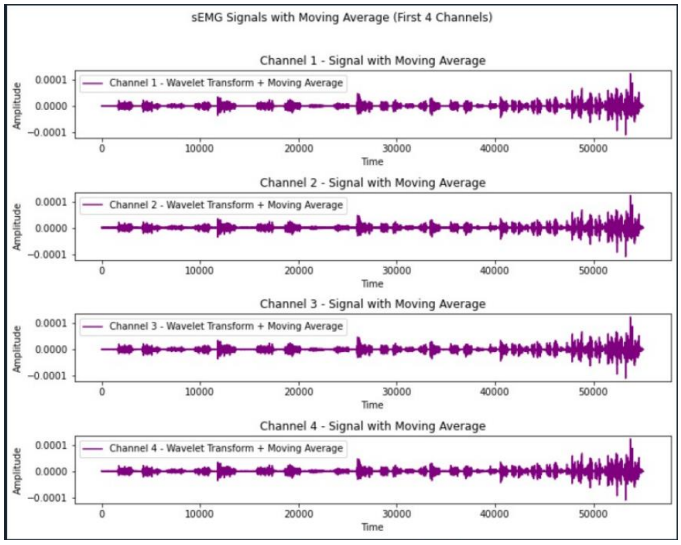
Inference:

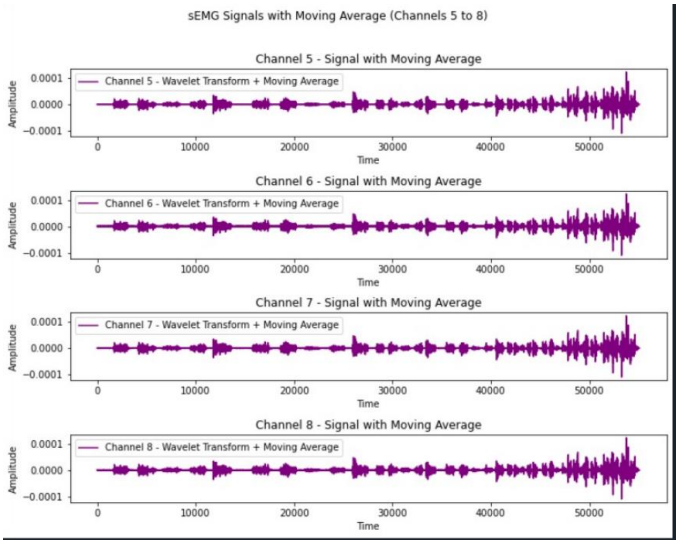
The K-Nearest Neighbors (KNN) classifier was trained on the preprocessed sEMG (surface electromyography) signals, which underwent bandpass filtering, Haar wavelet transformation, and moving average. The dataset was split into training and testing sets, standardized, and used to train the KNN classifier. The classifier achieved an accuracy of 0.98 on the test set. This high accuracy indicates that the model performed well in classifying gestures based on the sEMG signals.

The classification report provides more detailed information about the model's performance for each class, including precision, recall, and F1-score. However, since you requested a brief inference, the focus is on the overall accuracy, which is 0.98. In summary, the KNN classifier demonstrated excellent performance in classifying gestures from sEMG signals, achieving a high accuracy rate of 98%. This suggests that the combination of bandpass filtering, Haar wavelet transformation, and moving average successfully extracted relevant features for effective gesture classification.









V. Acknowledgment

The development of this project was greatly influenced by the work of various researchers and authors in the field of electromyographic (EMG) signal processing and machine learning. We would like to express our gratitude to the following researchers and institutions:

1.Mora Rubio, A.; Alzate Grisales, J.A.; Tabares-Soto, R.; Orozco-Arias, S.; Jiménez Varón, C.F.; Padilla Buriticá, J.I. for their contribution in "Identification of Hand Movements from Electromyographic Signals Using Machine Learning" Preprints 2020, 2020020443.

2.Baixin Sun, Guang Cheng, Quanmin Dai, Tianlin Chen, Weifeng Liu, Xiaorong Xu for their work on "Human motion intention recognition based on EMG signal and angle signal" Cognitive Computation and Systems.

3.The contributors to the "EMG data for gestures" dataset available at the UCI Machine Learning Repository.

Their pioneering work has significantly contributed to the advancement of EMG-based gesture recognition and pattern analysis, inspiring and informing the development of this project.

References

1. Mora Rubio, A.; Alzate Grisales, J.A.; Tabares-Soto, R.; Orozco-Arias, S.; Jiménez Varón, C.F.; Padilla Buriticá, J.I. Identification of Hand Movements from Electromyographic Signals Using Machine Learning. *Preprints* **2020**, 2020020443. https://doi.org/10.20944/preprints202002.0443.v1
Identification of Hand Movements from Electromyographic Signals Using Machine Learning[v1] | Preprints.org

2. Baixin Sun, Guang Cheng, Quanmin Dai, Tianlin Chen, Weifeng Liu, Xiaorong Xu
https://doi.org/10.1049/ccs2.12002

Human motion intention recognition based on EMG signal and angle signal - Sun - 2021 - Cognitive Computation and Systems - Wiley Online Library
3. Real-Time EMG Based Pattern Recognition Control for Hand Prostheses: A Review on Existing Methods, Challenges and Future Implementation - PMC (nih.gov)
4. Alexander J. Shackman, Brenton W. McMenamin, Heleen A. Slagter, Jeffrey S. Maxwell, Lawrence L. Greischar, and RichardJ.Davidson[1]https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2712576/
5. The MARCS Institute, School of Computing, Engineering and Mathematics, Western Sydney UniversityDepartment of Information Technology and Electrical Engineering, "Federico II" The University of Naples, School of Engineering, Macquarie University,Real-Time EMG Based Pattern Recognition Control for Hand Prostheses: A Review on Existing Methods, Challenges and Future Implementation - PMC (nih.gov)
6. Real time virtual prosthetic hand controlled using EMG signals | IEEE Conference Publication | IEEE Xplore
7. Ghofrani Jahromi, M.; Parsaei, H.; Zamani, A.; Stashuk, D.W. Cross Comparison of Motor Unit Potential Features Used in EMG Signal Decomposition. IEEE Transactions on Neural Systems and Rehabilitation Engineering 2018, 26, 1017–1025. doi:10.1109/TNSRE.2018.2817498
8. Gokgoz, E.; Subasi, A. Comparison of decision tree algorithms for EMG signal classification using DWT. Biomedical Signal Processing and Control 2015, 18, 138–144. doi:https://doi.org/10.1016/j.bspc.2014.12.005.
9. Singh, S.P.; Urooj, S. Wavelets: Biomedical applications. International Journal of Biomedical Engineering and Technology 2015, 19, 1–25. doi:10.1504/IJBET.2015.071405.
10. Guido, S.; C. Müller, A. Introduction to machine learning with scikit-learn; O'Reilly Media, Inc., 2016.
11. EMG data for gestures - UCI Machine Learning Repository

Appendix
Code

```
# -*- coding: utf-8 -*-
"""
Created on Mon Dec 18 09:52:17 2023

@author: kavya
"""

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score,
classification_report
from sklearn.preprocessing import StandardScaler
from scipy.signal import butter, filtfilt

# Load your dataset
# Replace 'your_dataset.txt' with the actual path or filename of
your text file
dataset =
pd.read_csv('C:/Users/kavya/OneDrive/Desktop/EMG_data_for_gestures-master/04/2_raw_data_18-03_24.04.16.txt',
delimiter='\t')

# Define a bandpass filter function
def butter_bandpass_filter(data, lowcut, highcut, fs, order=4,
min_padlen=3):
    nyquist = 0.5 * fs
    low = lowcut / nyquist
    high = highcut / nyquist
    b, a = butter(order, [low, high], btype='band')

    # Calculate a suitable padlen based on the length of the data
    padlen = max(min_padlen, len(data) + min_padlen)  #
Ensure padlen is at least the minimum

    # Pad the signal
    data_padded = np.pad(data, (padlen, padlen), mode='edge')

    # Apply bandpass filter
    y = filtfilt(b, a, data_padded)

    return y[padlen:-padlen]


def haar_wavelet_transform(signal):
    coeffs = []
    while len(signal) >= 2:
        if len(signal) % 2 != 0:
            signal = np.append(signal, 0)  # Pad with zero if the
length is odd
        avg = (signal[0::2] + signal[1::2]) / 2.0
        diff = (signal[0::2] - signal[1::2]) / 2.0
        coeffs.append(diff)
        signal = avg
    if len(coeffs) == 0:
        return np.array([])
    coeffs.append(signal)  # Approximation coefficients
    return np.concatenate(coeffs)

# Define a moving average function
def moving_average(data, window_size):
    return np.convolve(data, np.ones(window_size) /
window_size, mode='valid')

# Plot the original sEMG signals
fig, axes = plt.subplots(8, 1, figsize=(10, 12))
fig.suptitle('Raw sEMG Signals from 8 Channels')

for i in range(8):
    ax = axes[i]
    ax.plot(dataset.iloc[:, i + 1].values, label=f'Channel {i + 1}',
color='blue')
```

```
    ax.set_xlabel('Time')
    ax.set_ylabel('Amplitude')
    ax.legend()

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

# Apply bandpass filter and Haar wavelet transform
# Create subplots for filtered signals - First 4 channels
plt.figure(figsize=(10, 8))
plt.suptitle('Filtered sEMG Signals - Channels 1 to 4')

for i in range(4):
    # Apply bandpass filter
    filtered_signal = butter_bandpass_filter(dataset.iloc[:, i +
1].values, lowcut=20, highcut=500, fs=4000)

    plt.subplot(4, 1, i + 1)
    plt.plot(filtered_signal, label=f'Channel {i + 1} - Filtered
Signal', color='green')
    plt.title(f'Channel {i + 1} - Filtered Signal')
    plt.xlabel('Time')
    plt.ylabel('Amplitude')
    plt.legend()

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

# Create subplots for filtered signals - Last 4 channels
plt.figure(figsize=(10, 8))
plt.suptitle('Filtered sEMG Signals - Channels 5 to 8')

for i in range(4, 8):
    # Apply bandpass filter
    filtered_signal = butter_bandpass_filter(dataset.iloc[:, i +
1].values, lowcut=20, highcut=500, fs=4000)

    plt.subplot(4, 1, i - 3)
    plt.plot(filtered_signal, label=f'Channel {i + 1} - Filtered
Signal', color='green')
    plt.title(f'Channel {i + 1} - Filtered Signal')
    plt.xlabel('Time')
    plt.ylabel('Amplitude')
    plt.legend()

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

# haar wavelet tranformed signal
# Create subplots for Haar wavelet transformed signals - First
4 channels
# Haar wavelet tranformed signal
# Create subplots for Haar wavelet transformed signals - First
4 channels
plt.figure(figsize=(10, 8))
plt.suptitle('Haar Wavelet Transformed sEMG Signals from 8
Channels')

for i in range(4):
    # Apply bandpass filter
    filtered_signal = butter_bandpass_filter(dataset.iloc[:, i +
1].values, lowcut=20, highcut=500, fs=4000)

    # Apply Haar wavelet transform
    transformed_signal =
haar_wavelet_transform(filtered_signal)

    plt.subplot(4, 1, i + 1)
    plt.plot(np.arange(len(transformed_signal)),
transformed_signal, label=f'Channel {i + 1} - Wavelet
Transform', color='red')
    plt.title(f'Channel {i + 1} - Wavelet Transformed Signal')
    plt.xlabel('Time')
    plt.ylabel('Amplitude')
    plt.legend()

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

plt.figure(figsize=(10, 8))
plt.suptitle('Haar Wavelet Transformed sEMG Signals from
Channels 5 to 8')

for i in range(4, 8):
    # Apply bandpass filter
    filtered_signal = butter_bandpass_filter(dataset.iloc[:, i +
1].values, lowcut=20, highcut=500, fs=4000)

    # Apply Haar wavelet transform
    transformed_signal =
haar_wavelet_transform(filtered_signal)

    plt.subplot(4, 1, i - 3)
    plt.plot(np.arange(len(transformed_signal)),
transformed_signal, label=f'Channel {i + 1} - Wavelet
Transform', color='red')
    plt.title(f'Channel {i + 1} - Wavelet Transformed Signal')
    plt.xlabel('Time')
    plt.ylabel('Amplitude')
    plt.legend()

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()


# Apply moving average to the Haar wavelet transformed
signal - First 4 channels
plt.figure(figsize=(10, 8))
plt.suptitle('sEMG Signals with Moving Average (First 4
Channels)')

window_size = 10  # You can adjust the window size as
needed
```

```
for i in range(4):


    # Apply moving average to the transformed signal
    smoothed_signal = moving_average(transformed_signal,
window_size)

    plt.subplot(4, 1, i + 1)
    plt.plot(np.arange(len(smoothed_signal)), smoothed_signal,
label=f'Channel {i + 1} - Wavelet Transform + Moving
Average', color='purple')
    plt.title(f'Channel {i + 1} - Signal with Moving Average')
    plt.xlabel('Time')
    plt.ylabel('Amplitude')
    plt.legend()

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

# Apply moving average to the Haar wavelet transformed
signal - Next 4 channels
plt.figure(figsize=(10, 8))
plt.suptitle('sEMG Signals with Moving Average (Channels 5
to 8)')

for i in range(4, 8):

    # Apply moving average to the transformed signal
    smoothed_signal = moving_average(transformed_signal,
window_size)

    plt.subplot(4, 1, i - 3)
    plt.plot(np.arange(len(smoothed_signal)), smoothed_signal,
label=f'Channel {i + 1} - Wavelet Transform + Moving
Average', color='purple')
    plt.title(f'Channel {i + 1} - Signal with Moving Average')
    plt.xlabel('Time')
    plt.ylabel('Amplitude')
    plt.legend()

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

# ...

# Apply Haar wavelet transform to the entire dataset
frequency_domain_features = dataset.iloc[:,
1:9].apply(lambda x:
haar_wavelet_transform(butter_bandpass_filter(x.values,
lowcut=20, highcut=500, fs=4000)), axis=1)

# Remove rows with empty arrays (signals that caused issues)
frequency_domain_features =
frequency_domain_features[frequency_domain_features.apply
(len) > 0]

# Apply moving average to the Haar wavelet transformed
signal
window_size = 10  # You can adjust the window size as
needed
processed_features =
frequency_domain_features.apply(lambda x:
moving_average(x, window_size))

# Combine processed features
X = np.vstack(processed_features.values)
y = dataset.loc[frequency_domain_features.index,
'class'].values

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Standardize features
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Create a K-Nearest Neighbors (KNN) classifier
clf = KNeighborsClassifier(n_neighbors=5)  # You can adjust
the number of neighbors as needed

# Train the classifier
clf.fit(X_train, y_train)

# Make predictions on the test set
y_pred = clf.predict(X_test)

# Evaluate the performance
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)

print(f"Accuracy: {accuracy:.2f}")
print("Classification Report:")
print(report)
```