Steady state Visual Evoked Potentials

Summer 2024
Digital Signal Processing
Dr Seif El-Dawlatly

Moaz Hafez – 900214137
Mostafa Elfaggal – 900211128
Nadia Dessouky – 900212701

Overview

This report presents the implementation of two approaches for Steady-State Visual Evoked Potential (SSVEP) classification using EEG data from two subjects. The methods involve preprocessing, CAR filter, component extraction, and classification using rule-based system once, and another time using K-Nearest Neighbors (KNN) method. The report includes the approaches used, the outcomes achieved, and the analysis of the results.

The Approach

Data Import and Initial Analysis

1. **Import Data as Numpy Arrays**:
    - Load EEG data and stimulus label data from CSV files into numpy arrays for both subjects.
2. **Data Analysis**:
    - **Findings**:
        - The number of sample entries for each stimulus frequency is 641 (128 * 5 + 1).
        - Resting periods (stimulus = 0) mostly consist of 767 entries (128 * 6 - 1).
        - These entries indicate a 5-second interval for stimuli and a 6-second interval for resting periods.

Data Restructuring

3. **Remove Resting Parts**:
    - Filter out the resting periods (where stimulus = 0) from both the EEG and stimulus data.

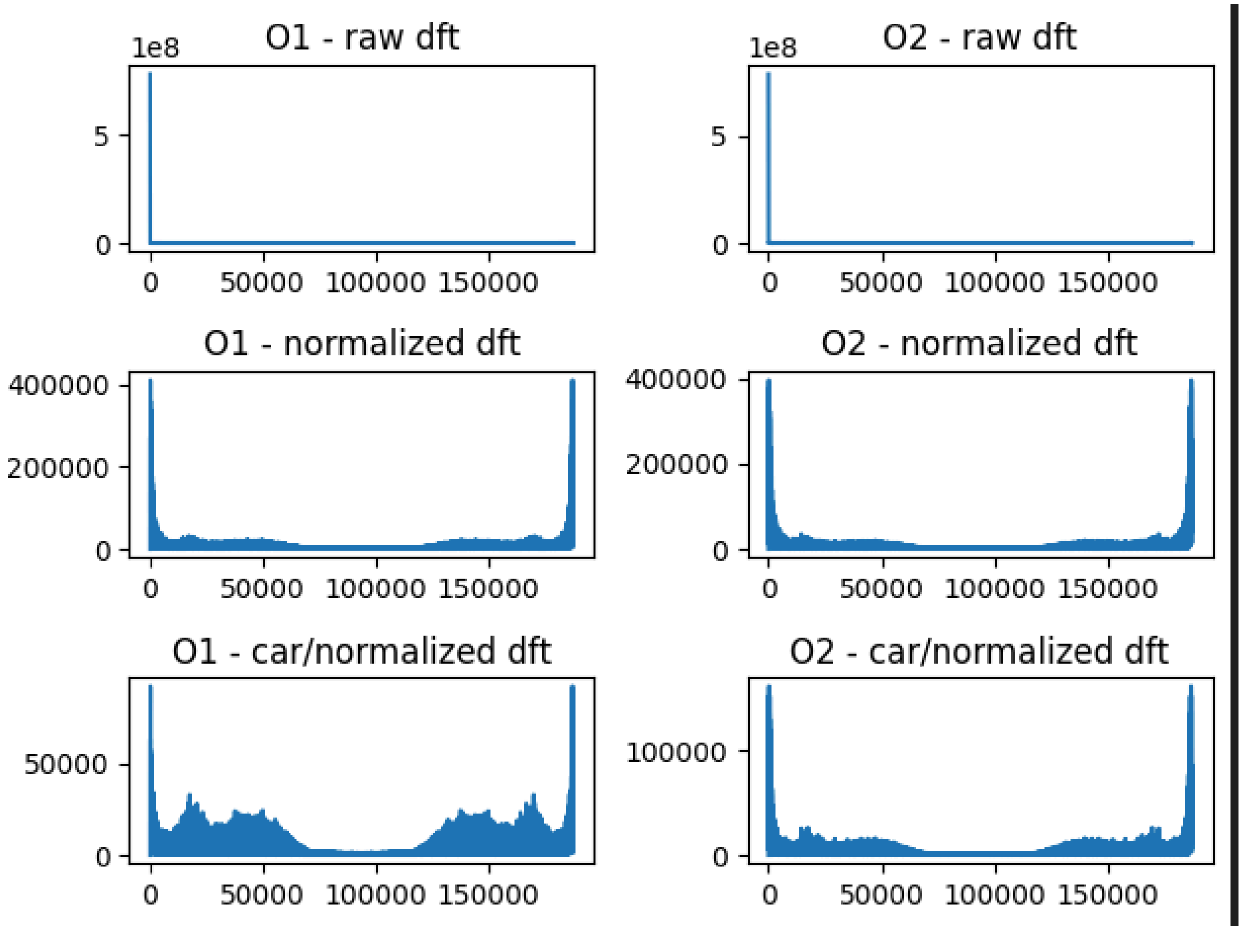
Data Processing and Normalization

4. **Normalize Data**:
   - Normalize the data to eliminate the DC offset in the frequency domain.
   - Achieved by:
     - Calculating the mean of each row.
     - Subtracting the mean from the corresponding row elements.
5. **Apply Common Average Reference (CAR)**:
   - This eliminates common noise across all electrodes vertically.
   - Achieved in a similar way to normalization but vertically across all rows

6. **Apply Discrete Fourier Transform (DFT)**:
   - Perform DFT on the filtered EEG data to transform it from the time domain to the frequency domain.

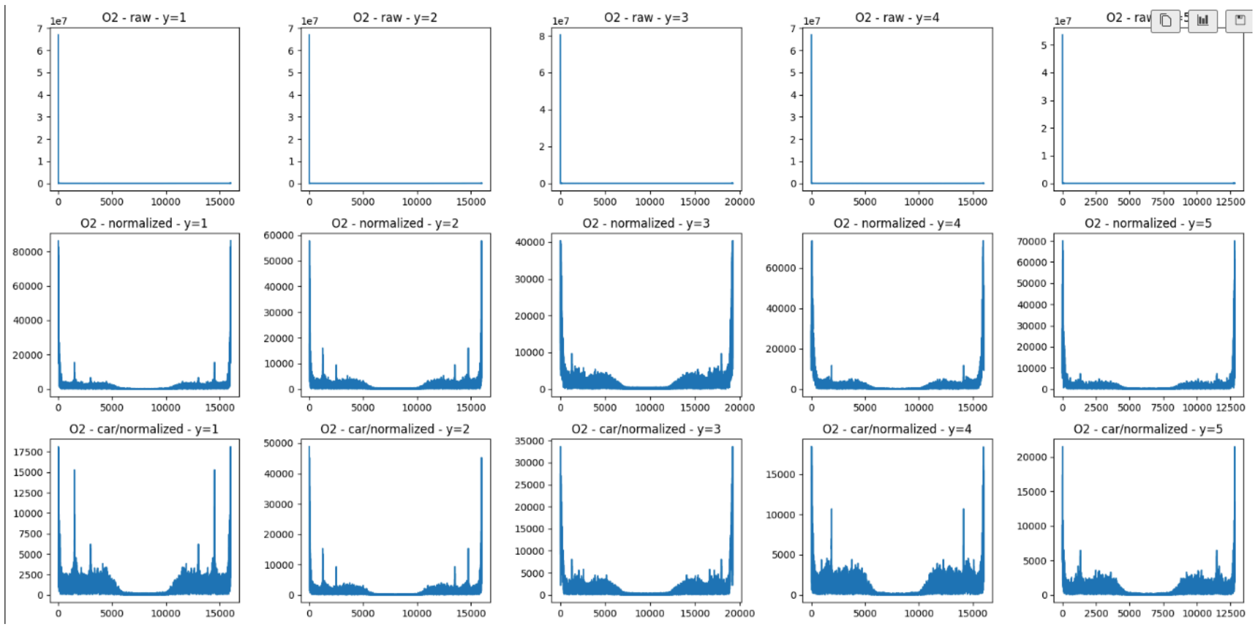
Results of DFT

Shape of the filtered data



The raw DFT data without filtering obtains a spectrum of frequencies with high dominance of low (near-zero) frequencies, leaving the remaining of the frequencies indistinguishable from the graphs.
The normalized frequencies removes the DC offset and hence removes the dominance of the low frequencies. The rest of the frequencies are hence became more distinguishable.
Adding the final stage of CAR filter removes the common noise across all electrodes. This reduces the magnitude of the frequencies lower, and makes the difference in the magnitude even more distinguishable.

Now the data is ready to be extracted and compared.



The data for each stimulus period.

Further Data Restructuring

7. **Aggregate Stimulus Entries**:
   - Group the 641 sample labels for each trial's stimulus frequency into a single entry.
   - This restructuring results in 125 entries (25 trials for each box).
8. **Reshape Data**:
   - Convert the data into a 3D array with the dimensions:
     - 125 trials
     - 14 electrodes
     - 641 frequency samples

DFT Extraction

9. **Extract frequency components**:

- Now that the data is transformed into the frequency domain, it remains to extract it from the array of DFT.
- This is achieved by
    - Searching for the index of the stimulus frequency we're searching for
    - Extracting these frequencies at the found index
    - (Method I) Find the maximum value of these frequencies to determine which is the desired output
    - Compare the output of the extraction method with the original stimulus data to obtain the results and the accuracy

```python
print(np.fft.fftfreq(N, d=1/f))
# 12 -> 60,61
# 10 -> 50, 51
# 8.57 -> 42, 43
# 7.5 -> 37, 38
# 6.66 -> 33, 34
```

✓ 0.0s

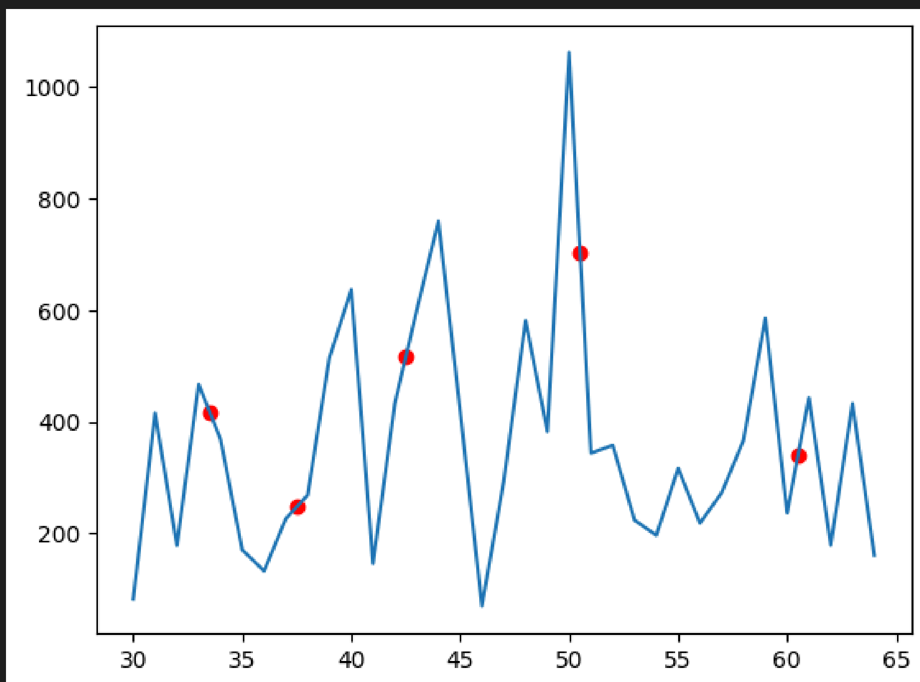
```
[ 0.          0.19968799  0.39937598  0.59906396  0.79875195
  0.99843994  1.19812793  1.39781591  1.5975039   1.79719189
  1.99687988  2.19656786  2.39625585  2.59594384  2.79563183
  2.99531981  3.1950078   3.39469579  3.59438378  3.79407176
  3.99375975  4.19344774  4.39313573  4.59282371  4.7925117
  4.99219969  5.19188768  5.39157566  5.59126365  5.79095164
  5.99063963  6.19032761  6.3900156   6.58970359  6.78939158
  6.98907956  7.18876755  7.38845554  7.58814353  7.78783151
  7.9875195   8.18720749  8.38689548  8.58658346  8.78627145
  8.98595944  9.18564743  9.38533541  9.5850234   9.78471139
  9.98439938 10.18408736 10.38377535 10.58346334 10.78315133
 10.98283931 11.1825273  11.38221529 11.58190328 11.78159126
 11.98127925 12.18096724 12.38065523 12.58034321 12.7800312
```

The function fftfreq() returns a 1D array of all the extracted frequencies of the DFT. It's found that the frequencies that we're searching for have the indicated indices in the array above.

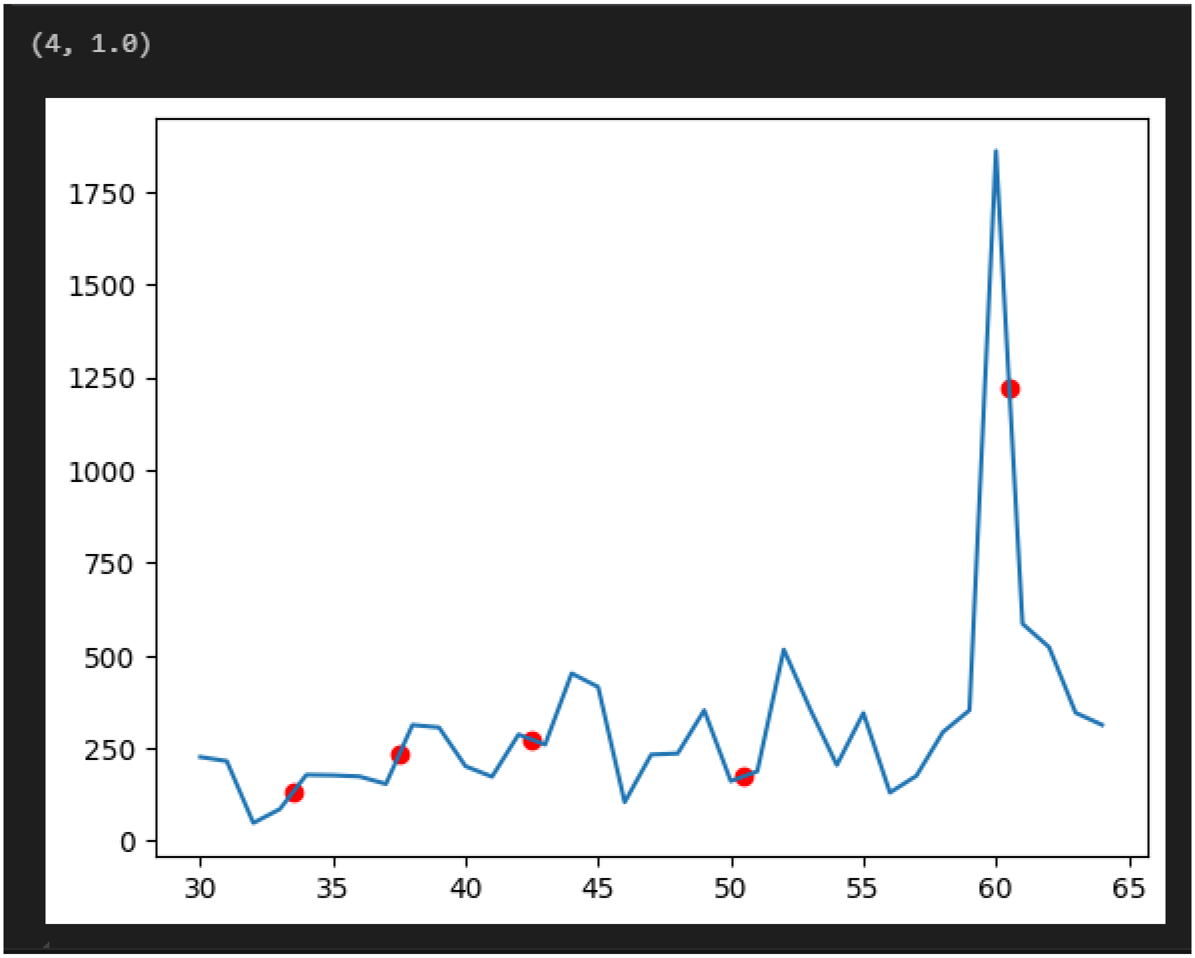
```python
trial_num += 1
window = (30, 65)

plt.plot(list(range(window[0],window[1])), trials_dft[trial_num, 7, window[0]:window[1]])
plot_components(get_components_val_from_dft(trials_dft[trial_num,7]))
trial_num, aggregate_y[trial_num]
```

✓ 0.1s

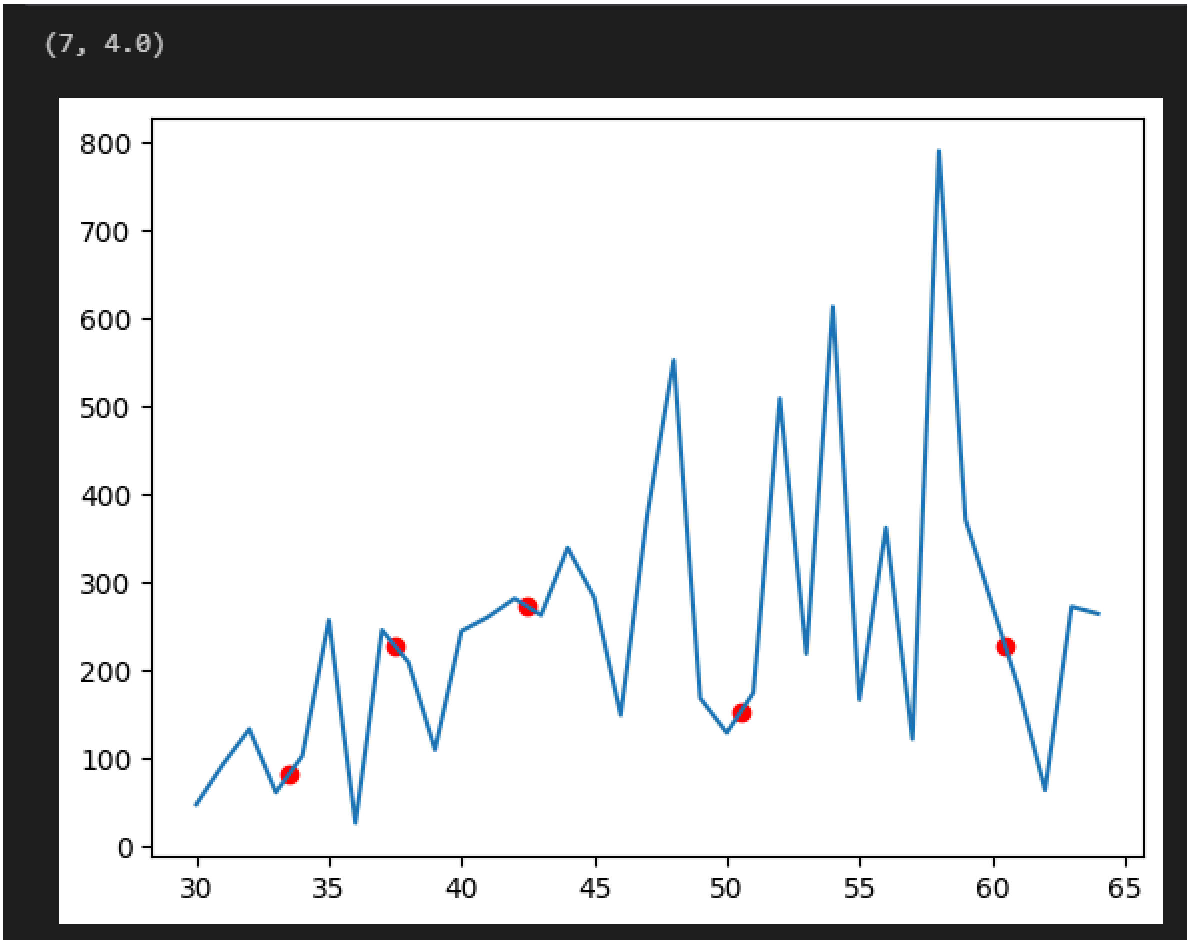
```
(1, 2.0)
```



The data of each trial is plotted, with focus on only the desired frequencies. The Figure above shows the frequency of indices around 50 to be the highest frequency, which corresponds to stimulus 2. This is a correct interpretation as the desired output was 2.



Similarly, the output is the frequency of 12 Hz (Stimulus 1) and the desired output was 1.



Here the results are faulty, the desired output is 4 while the output of the model was 3.

Now remains to find the exact classification accuracy of the model
This is done by comparing the extracted frequency with the expected output y for each of the 125 trials

Method I

Subject 1

```
O1: 54.400000000000006 % | O2: 65.60000000000001 % | O_avg: 61.6 %
```

Average method is suggested for combination of the two electrode data.

Subject 2

```
O1: 48.0% | O2: 84.0% | O_avg: 74.4%
```

Method II

Data Processing

Similarly, the KNN method is processed from the CAR normalized data
The focus is on optimizing the KNN classifier by selecting the best k value and component window size to achieve maximum classification accuracy.

This is achieved by

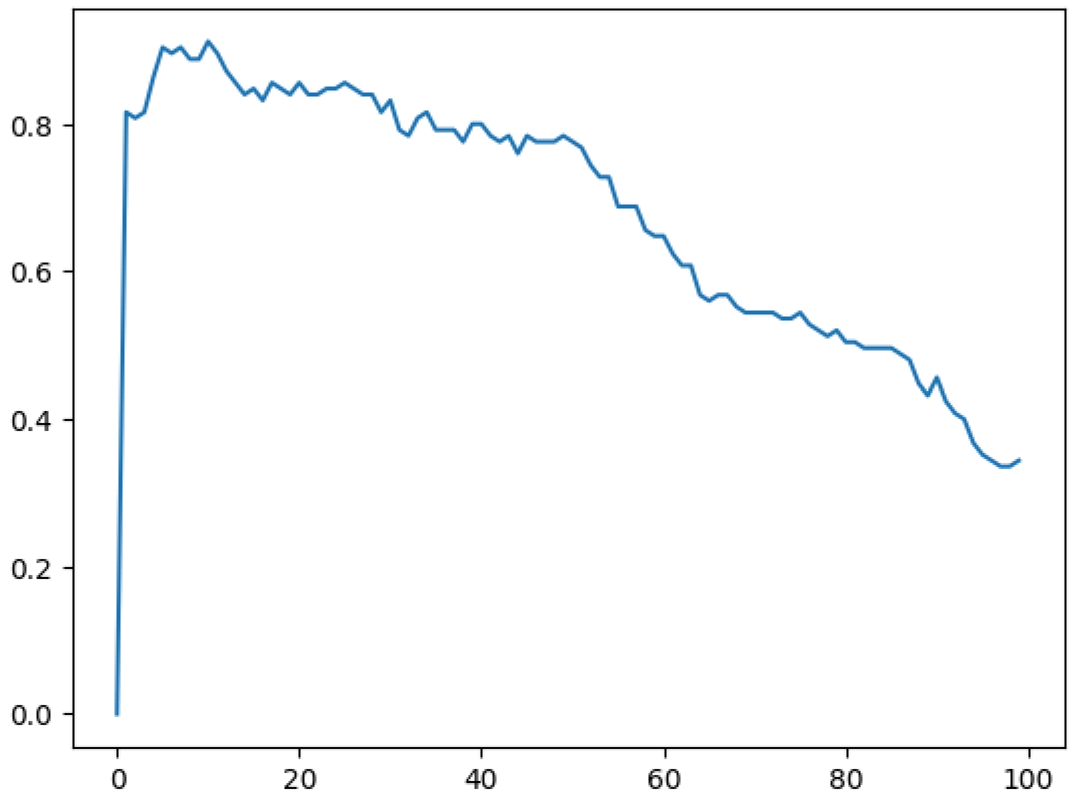
- **Best Window Search:** the window size affects the accuracy of the KNN model. Although the desired frequencies lie in the dft’s 30-65. Better accuracies can be attained with other window ranges. To find the best window range, we applied a brute force search to look through all possible window ranges and find the best one. Some pruning was needed to minimize the spectrum of windows tested without affecting the result.
    - The window start shouldn’t exceed 30 and the window end shouldn’t be less than 65 to include all the range of frequencies we’re searching for.
    - The window end shouldn’t be greater than 320 because of the symmetry property of the signal.

- **Early Stopping Criteria**:
    - After computing at least 10 windows, check if the mean accuracy over the last 5 windows is significantly lower than the current accuracy. If so, break the loop for the current window_start as further computations may be unnecessary.

- If the accuracy over the last 10 windows is equal to the current accuracy, break the loop as upcoming windows are likely not to be any better.

Best K Analysis:

K shouldn't be too small nor too large. A very small K would lead to the outliers affecting the prediction. While a very large K would lead to it being affected by the majority population.



Results:
Subject 1

O1 S1

```
Start: 26.0 | End: 101.0 | K: 24.0 | Acc: 88.8
```

O2 S1

```
Start: 26.0 | End: 101.0 | K: 24.0 | Acc: 88.8
```

O1, O2 Avg S1

```
Start: 17.0 | End: 143.0 | K: 23.0 | Acc: 91.2
```

O1, O2 Concat S1

```
Start: 29.0 | End: 138.0 | K: 22.0 | Acc: 91.2
```

14 Electrodes S1

```
Start: 9.0 | End: 88.0 | K: 23.0 | Acc: 80.80000000000001
```

Subject 2

O1 S2

```
Start: 29.0 | End: 138.0 | K: 22.0 | Acc: 91.2
```

O2 S2

```
 Start: 5.0 | End: 187.0 | K: 20.0 | Acc: 96.0
```

O1,O2 Avg S2

```
Start: 17.0 | End: 143.0 | K: 23.0 | Acc: 91.2
```

O1,O2 Concat S2

```
Start: 29.0 | End: 138.0 | K: 22.0 | Acc: 91.2
```

14 Electrodes S2

```
Start: 28.0 | End: 153.0 | K: 22.0 | Acc: 82.39999999999999
```

Best Window for

O1 -> 27 : 119

O2 -> 5 : 187

O1,O2 Avg 17 : 143

O1,O2 Concat 29 : 138

Conclusion

In this project, we implemented and evaluated two methods for classifying Steady-State Visual Evoked Potentials (SSVEP) using EEG data from two subjects: a rule-based approach and a K-Nearest Neighbors (KNN) classifier. Both methods involved preprocessing steps such as normalization, Common Average Reference (CAR) filtering, and Discrete Fourier Transform (DFT) application. The rule-based method focused on extracting frequency components and comparing them to the stimulus data, while the KNN method optimized classification through parameter tuning. The KNN classifier demonstrated higher accuracy and robustness compared to the rule-based approach. Our findings suggest that machine learning techniques, especially KNN, can significantly enhance SSVEP classification accuracy, offering promising potential for further research and application.