Project Report
CSE-572 Data Mining
Spring -2018

Submitted to
Prof. Ayan Banerjee
IRA A Fulton School of Engineering
Arizona State University

Submitted By

Venkata Ravi Teja Reddy Yanamala (1213420206)

Sai Nageswara Koushik Dhumantarao (1213158581)

Shreyas Shrinivas Mugali (1212904028)

Atul Mishra (1213363981)

Yeswanth Chowdary Narra (1213523673)

Table of Contents

1. Introduction:

This project is part of the Data Mining course taken by Dr. Ayan Banerjee in the Spring of 2018. The project solves the issue of a machine-level understanding of gesture recognition with the help of data from different sensors. The gestures are taken from a standard source American Sign Language and we need to analyze the input data from sensors to distinguish between 10 different gestures- About, And, Can, Cop, Deaf, Decide, Father, Find, Go Out, Hearing.

2. Team Members:

Venkata Ravi Teja Reddy Yanamala
Sai Nageswara Koushik Dhumantarao
Shreyas Shrinivas Mugali
Atul Mishra
Yeswanth Chowdary Narra

3. Assignment 1

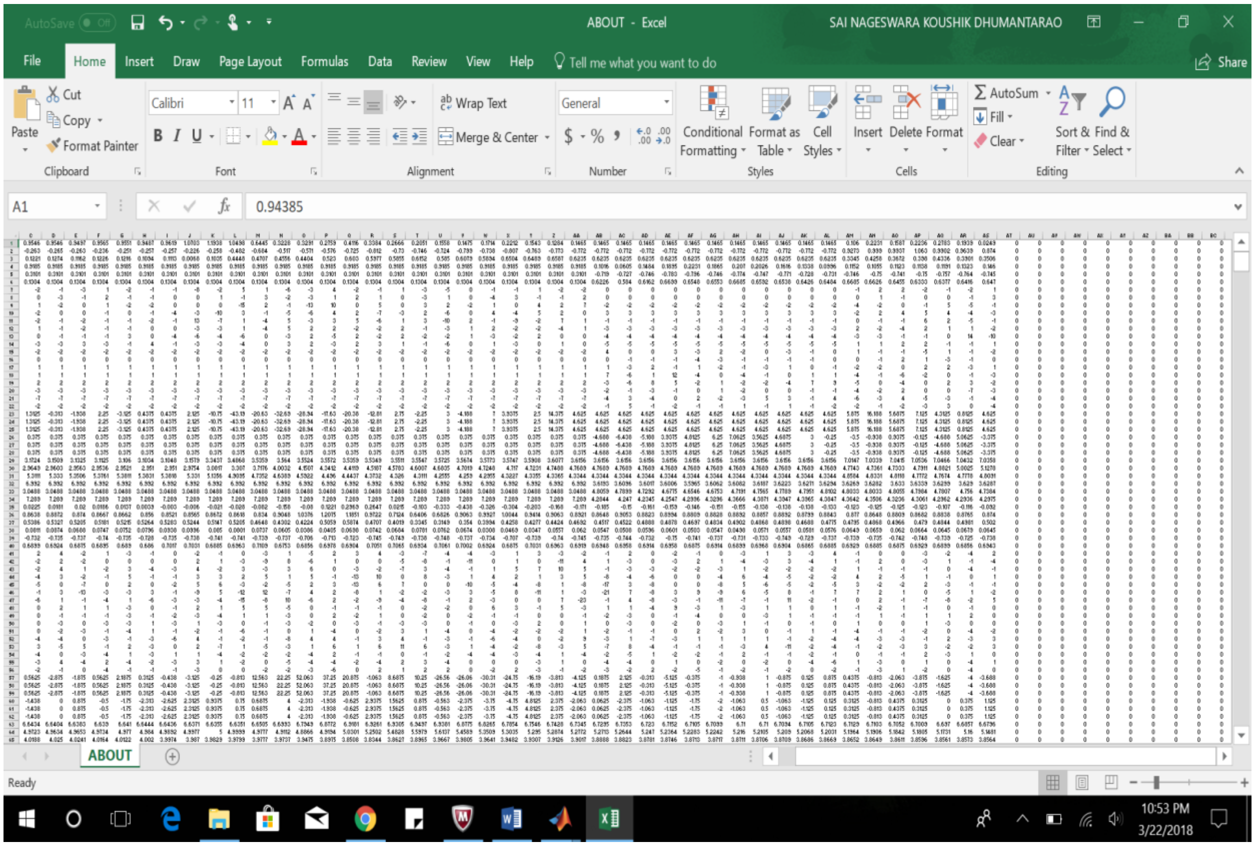
In Assignment 1 we collected data. One person from the team had to wear the wristband sensors and enact the gesture in front of the camera. Each gesture was enacted in an interval of 3 seconds and repeated for about 20 times, also there were 37 teams collecting the data via 34 sensors. So, ideally, we should have a total of approx. 20*37*34 (assume N for later references) input sensor data rows for each of the 10 gestures.

4. Assignment 2

The second assignment comprises of the following three tasks to analyze the given gestures: - Annotation, Feature Extraction Method, Feature Selection Method

4.1. Task 1: Annotation

In Assignment 2 Task 1 we had to annotate and segment the raw data. For this we used a MATLAB code to store each time series of an action column-wise and each row indicates a given sensor. We then append multiple actions of the respective gesture in the same file. Finally, we could segment the raw data in 10 different csv files corresponding to the 10 different gestures. Here, we need to note that all the input data does not have equal number of time series columns. This is due to errors while calibrating the sensors or human error where a person would start doing the gesture a little late. To standardize all the data values, we have zero-padded each row to a fixed value of 55 columns. If any row in an action file has more than 55 columns of time series then we have ignored that file and moved ahead to the next file. From here on we are using these zero-padded matrices to do feature extraction which is described in the next section. Following is a screenshot of the annotation file for the gesture- About which has 55 time series columns.



4.2. Task 2: Feature Extraction

For this task we used the following 5 different feature extraction methods to categorize the gesture data behavior-

- Fast Fourier Transform
- Mean
- Variance
- RMS
- Discrete Wavelet Transform

In this section first, we will provide a brief description of each of the above methods then we will state the underlying intuition that we used to identify a gesture and finally we will go on to show and justify the analysis results that we got by applying the feature extraction methods in MATLAB.

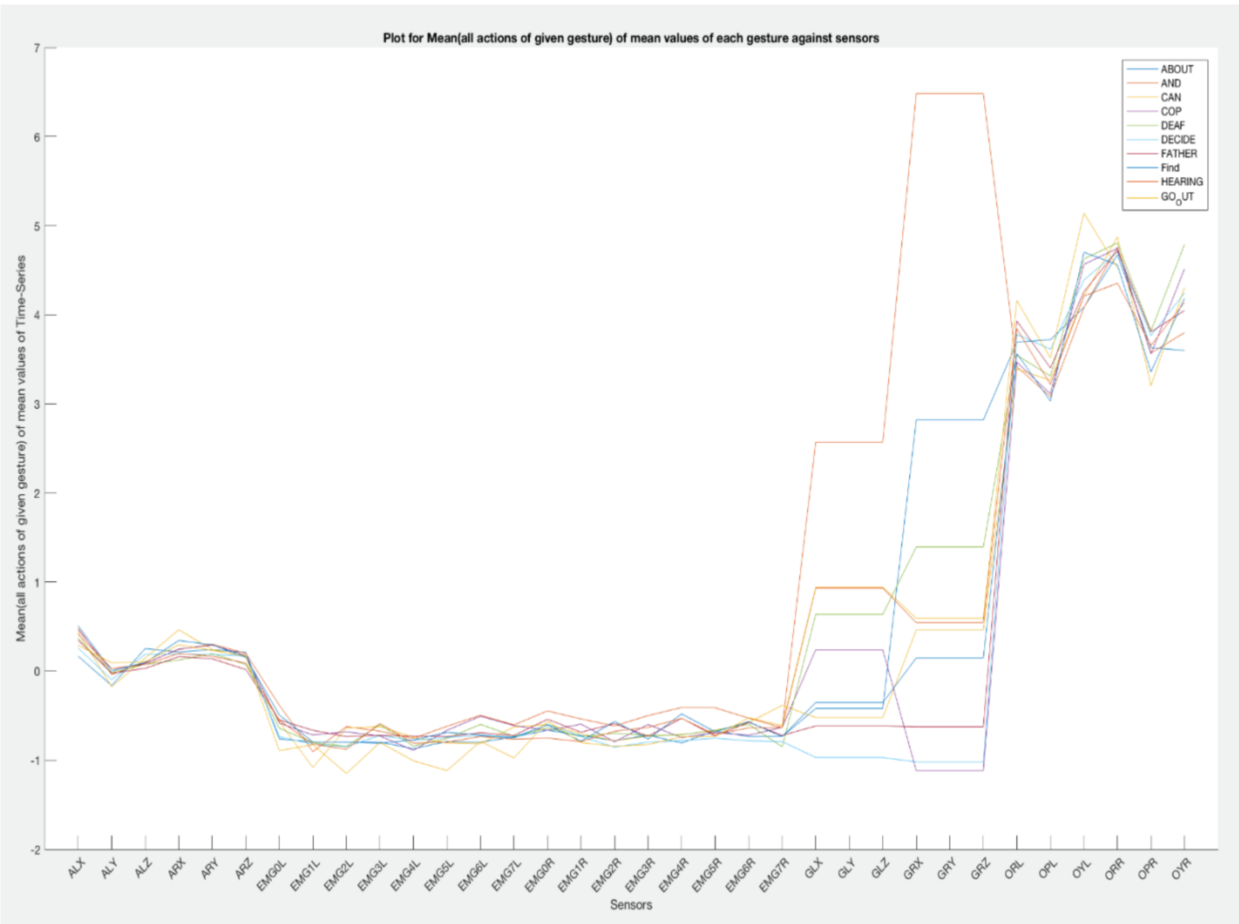
4.2.1. Mean

a) The mean function gives the mean value of all the time series data in each row of the raw data matrix. Here, we are combining the mean value of each row from all the actions for each sensor and then calculating the mean of all the means. So, finally we get the mean of mean value of each sensor for all the 10 gestures which is then combined and plotted in one graph. We used the *mean* function in the MATLAB library to achieve this feature extraction.

b) The intuition behind this method is that the Gyroscope and Orientation sensors will show some interesting result as the gestures performed have different orientation and angular velocity along the 3-D space. There should be identifiable distinctions along all the axis X, Y and Z for both the sensors.

c) MATLAB function name: findMean()

d) The graph plot for mean of all the sensor for different gestures is shown below. Here, X-axis is the sensor and Y-axis gives the mean value for the sensor.



e) As we can see in the above plot the mean values of all the gyroscope and orientation sensors are pretty high as compared to the other sensors. So, our intuition that these two sensors will show some interesting result comes true. One major thing to note from the graph is that the gyroscope sensors show a greater distinctness than the orientation sensors i.e., the mean values are significantly far from each other in the gyroscope sensors. This proves our intuition that the gyroscope sensor values can distinguish the gestures easily.

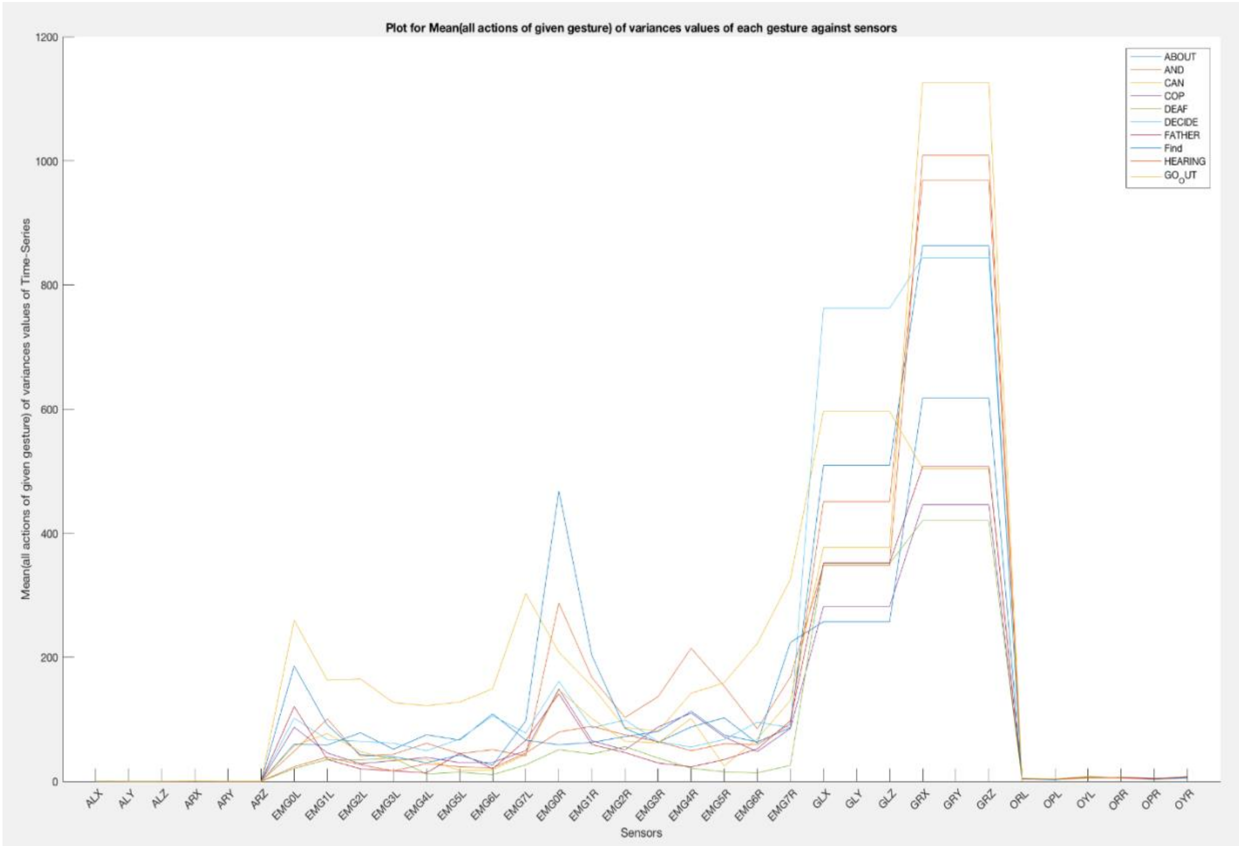
4.2.2. Variance

a) The variance function gives the divergence of all the elements in the data matrix. Here, we are taking all the rows of a sensor and performing variance along the row and then we take a mean of all the variance values of that sensor. This is done for all the sensors and all the corresponding gestures. Finally, we plot the mean of the variances for all sensors and all gestures. We used the *var* function in the MATLAB library to achieve this.

b) The intuition here is that we expect to see different levels of variance for the gyroscope or orientation sensor. The Orientation Sensor measures three values. The Azimuthal, pitch and the roll in degrees. The Gyroscope is used to find the angular velocity on all three axes. It has a value only when there is motion among any of the three values. Since most of the gestures would involve the change in the velocity along either the azimuthal, roll and pitch, gyroscope and orientation would be a good sensor to distinguish between various gestures.

c) MATLAB function name: findVar()

d) The graph plot for variance of all the sensor for different gestures is shown below. Here, X-axis is the sensor and Y-axis gives the variance value for the sensor.



e) Our intuition regarding either the orientation or gyroscope sensor having a larger variance is correct as demonstrated by the results above. The variance along the gyroscope sensor can be used to distinguish between different gestures.

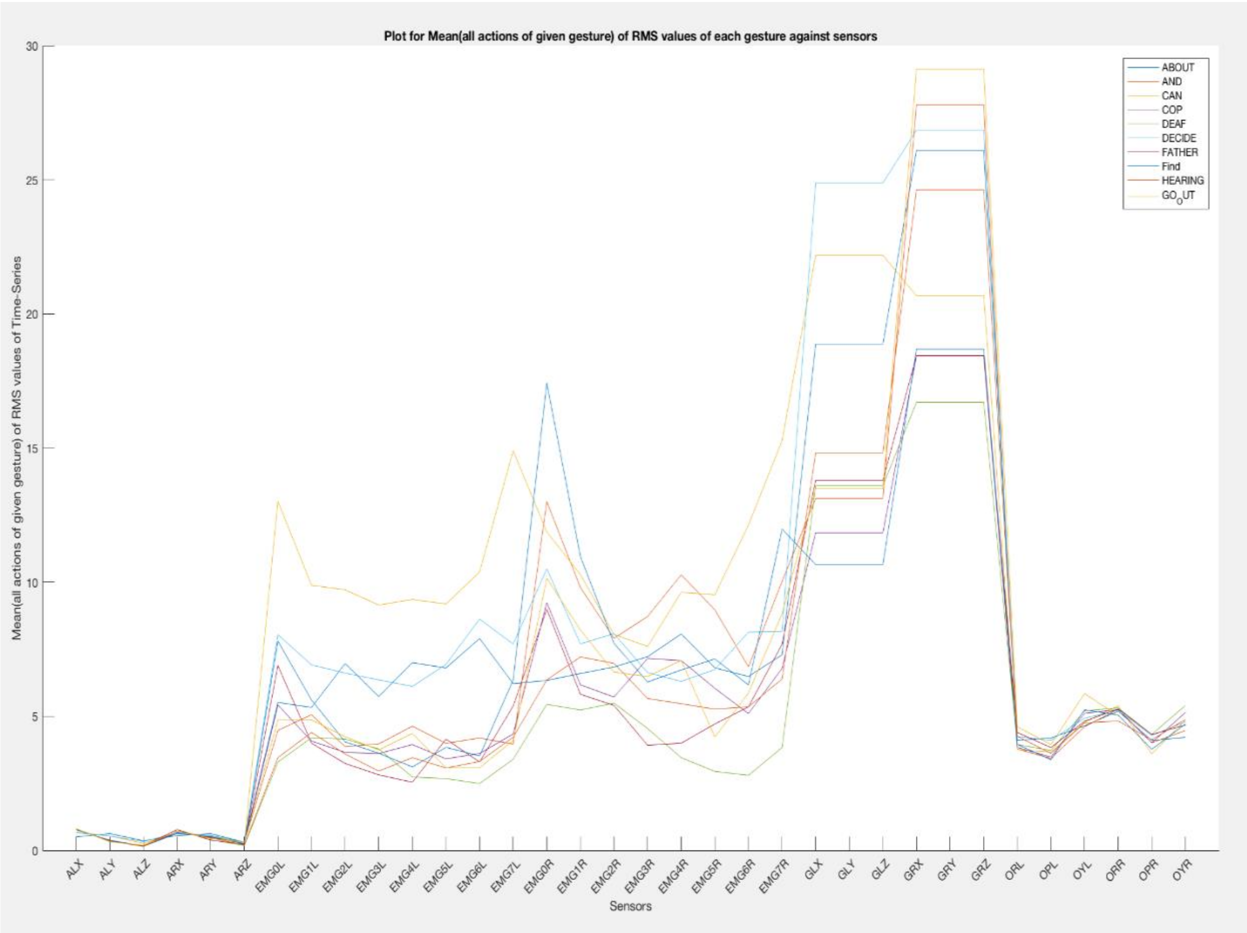
4.2.3. RMS

a) The RMS function gives the root mean square value of all the time series data in each row of the raw data matrix. Here, we are combining the rms value of each row from all the actions for each sensor and then calculating the mean of all the rms values. We used the *rms* function in the MATLAB library to achieve this feature extraction. Finally, the mean of the rms value of each sensor for all the 10 gestures are combined and plotted in one graph.

b) EMG sensors record the electrical activity produced by the skeletal muscles. Each gesture might require different level of muscular excitation. Our intuition is that RMS would somehow capture the power content of the time series and that these values of the EMG sensor data would help differentiate the gestures.

c) MATLAB function name: findRMS

d) The graph above represents the mean of rms values. The sensors are represented by the X-axis and Y-axis represents the magnitude of mean of the rms values.



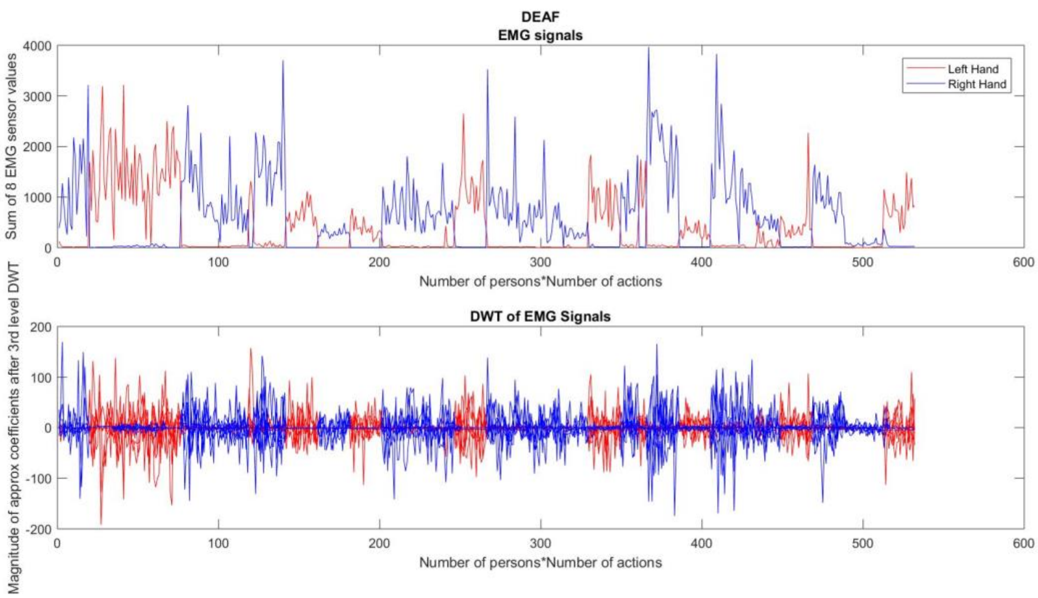
f) Our intuition proves to be True: From the graph, It can be seen that the EMG and gyroscope sensors have high magnitude compared to other sensors and the significant difference between the EMG sensor values between the gestures help us differentiate each gesture from the other therefore validating our intuition.

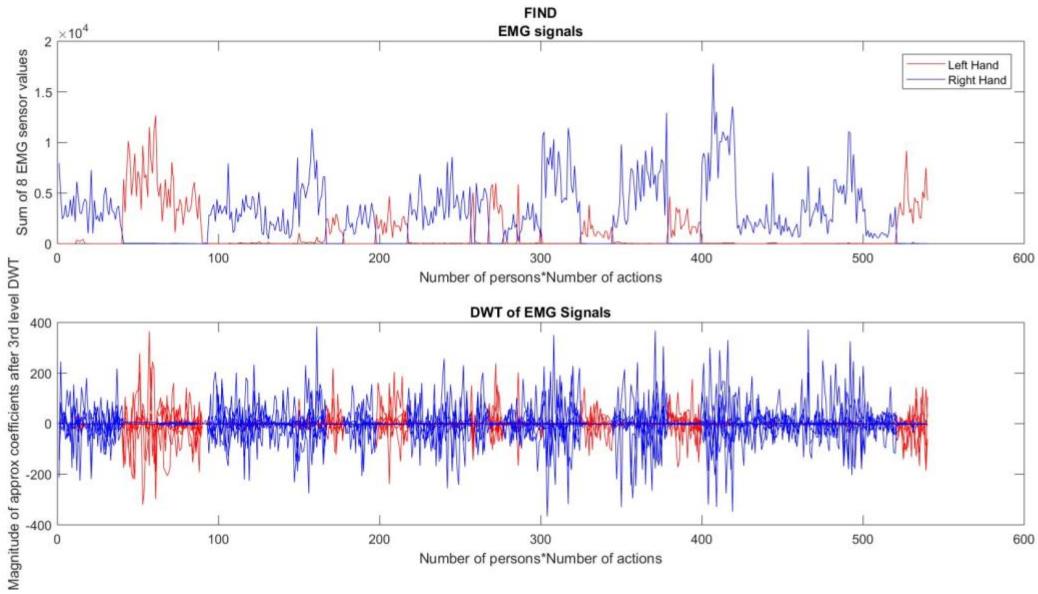
4.2.4. Discrete Wavelet Transform(DWT)

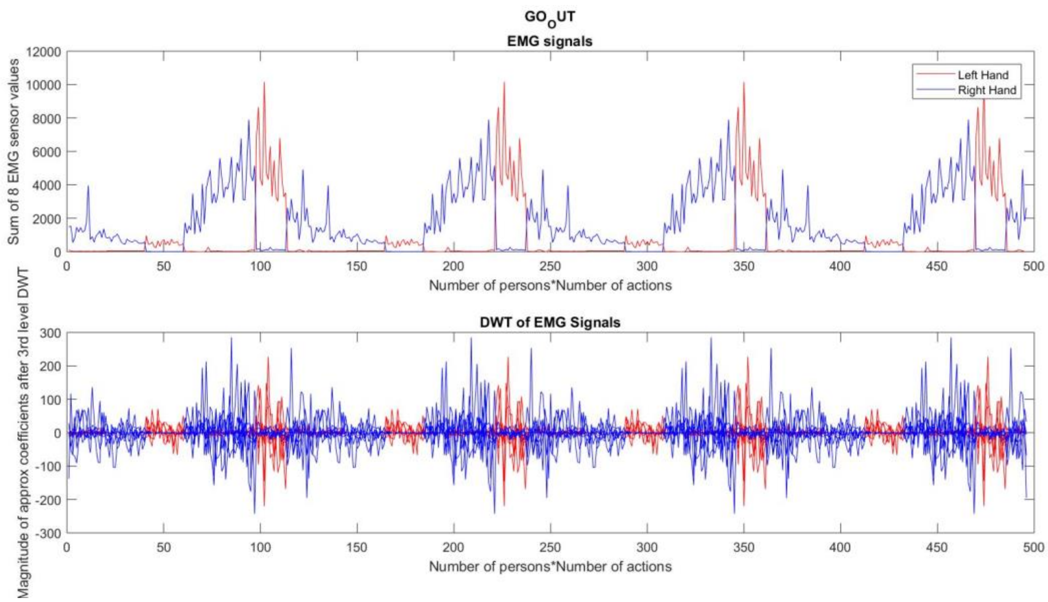
a) The discrete wavelet transform function decomposes the raw sensor data into wavelets. Here, we are decomposing the EMG sensor data to look for some interesting results. We used the dwt function in the MATLAB library to transform the data into discrete wavelets. We summed up the EMG sensor data for left hand and right hand separately for each action of the person. DWT is applied to both the EMG left sensor and EMG right sensor data separately using the dwt function in wavelet toolbox of MATLAB. So, each action of the person will have 2 rows, one row for the EMG left sensor and the other for EGM right sensor. The DWT function gives approximate (low frequency) and detail coefficients (high frequency). We use level 3 DWT. The first level detail coefficients are then given as an input to the second level DWT and the same is repeated for the third level. The approximate coefficients show the different behavior of the gestures over a longer period as compared to detail coefficients which gives the sudden behavioral changes. We use approximate DWT level 3 coefficients, as all the gestures will have major differences in behaviors over the full period in which they are performed.

b) Electromyography (EMG) sensor measures and records the electrical activity associated with muscle contractions. Since all the gestures involves some unique muscle activity, we expect to see differences that can be used to distinguish between gestures by applying DWT.

c) MATLAB function name: findDWT()
d) Result screenshots and explanation







e) The graph above depicts both the original and reconstructed signal using approximate coefficients from DWT. Using the level 3 approximate coefficients from DWT, we were able to reconstruct the signal which is similar to the original signal. Hence, we take the approximate coefficients into consideration while forming the feature matrix. The gestures have major differences in the approximate DWT coefficients. So, our intuition regarding using DWT on EMG sensors is validated.

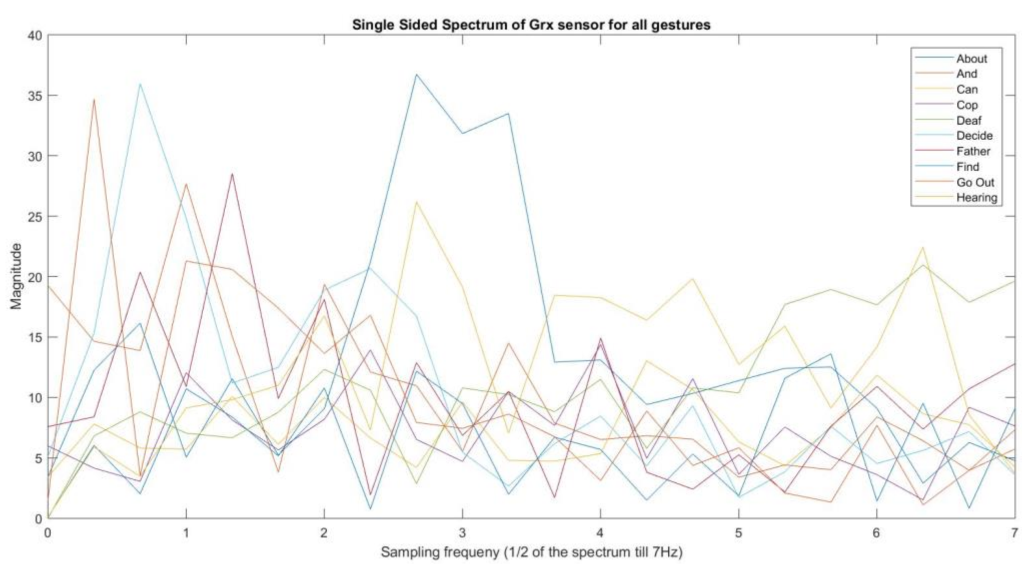
4.2.5. Fast Fourier Transform

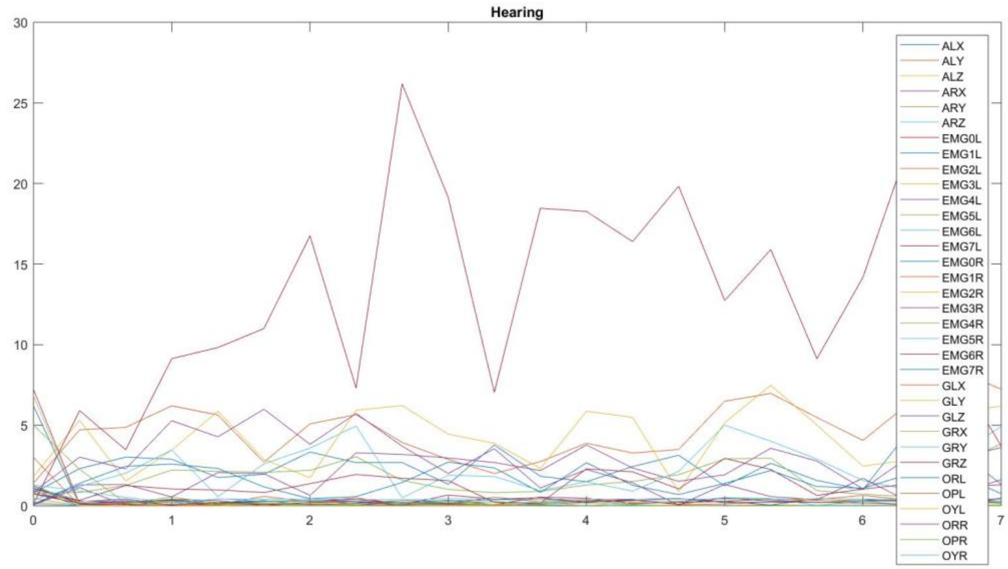
a) The FFT function gives the frequency domain representation of all the time series data in each row of the raw data matrix. Here, we are taking the sensor data and are calculating a two-sided spectrum, then from the two-sided spectrum we calculate a single sided spectrum, multiplying its magnitude by 2 along with the even-valued signal length. We then, consider the top 5 FFT coefficients value of each sensor from all the actions. For this particular feature extraction, We used the *fft* function from the MATLAB library.

b) During the process of data collection, every gesture has a different way of movement in terms of Orientation, speed. Gyroscope sensors provide the angular velocity along the orientation and since several gestures rely heavily on orientation, our intuition is that gyroscopic values will help in differentiating the gestures.

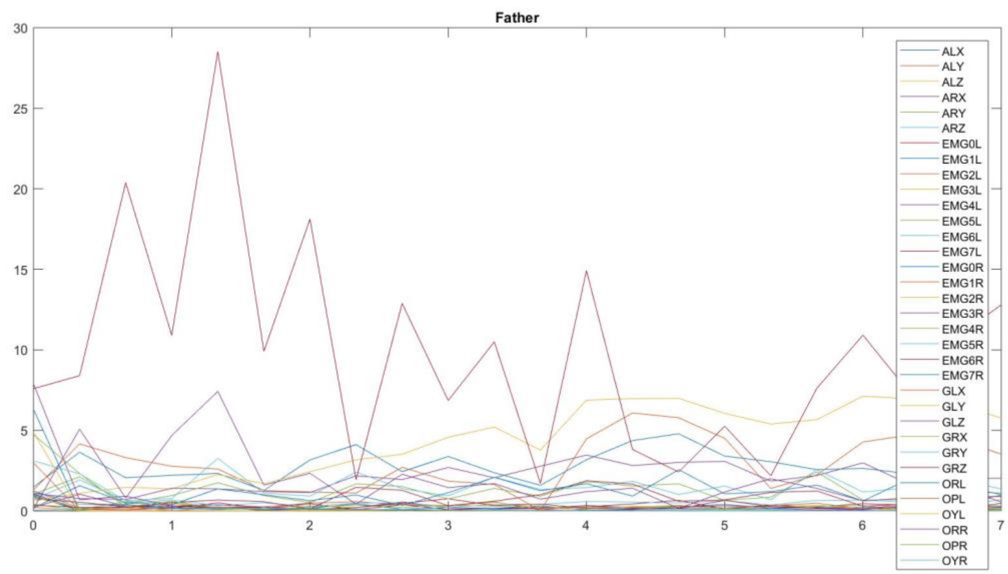
c) MATLAB function name: findFFT()

d) The first graph below shows the single sided FFT spectrum of gyroscope for all the gestures. The remaining graphs show that spectrum of gyroscope is more spread out.





X-axis represents one-half of the sampling frequency (7Hz); Y-axis represents the Magnitude



X-axis represents one-half of the sampling frequency (7Hz); Y-axis represents the Magnitude

e) As seen from the graph, The FFT values for the gyroscope are more spread out and can be used for distinguishing different gestures. The FFT values for other sensors are close together and hence cannot be used as a distinguishing parameter. Hence our intuition regarding gyroscope being the differentiator holds true.

4.3. Task 3: Feature Selection Method

In this task we need to analyze the different feature extraction methods to see which one gives the most variance using Principal Component Analysis.

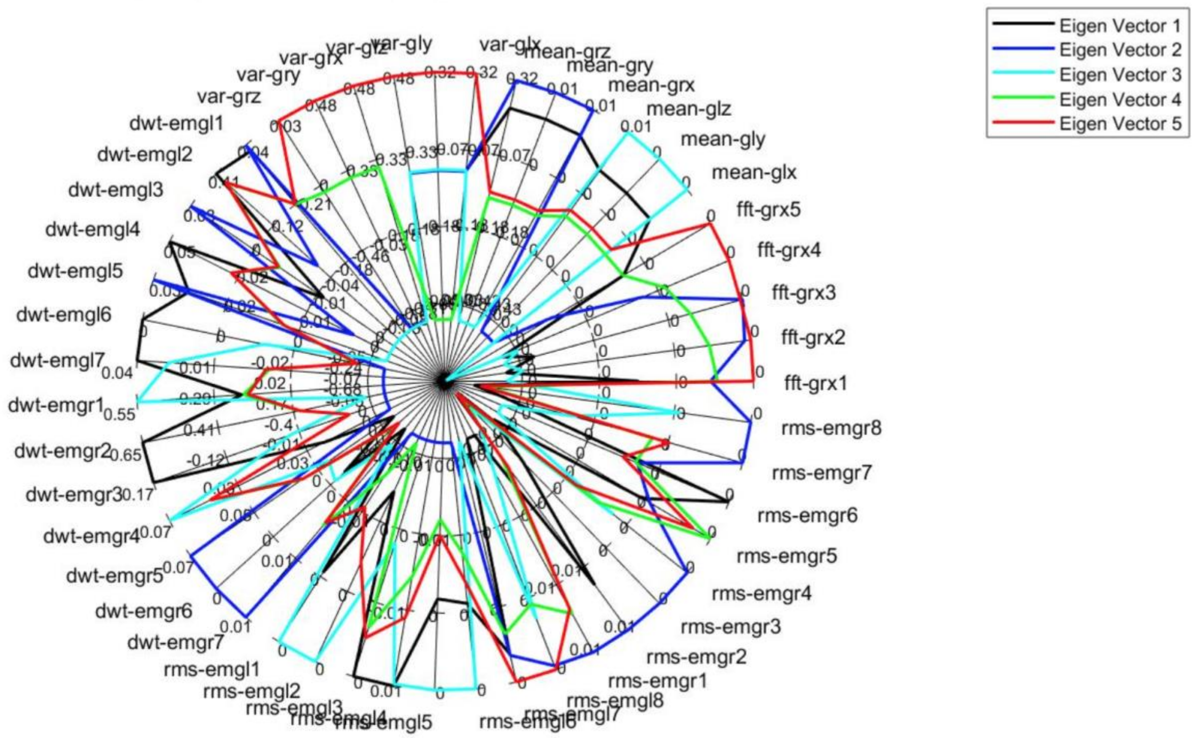
4.3.1. Data Feature Matrix

For this subtask we need form a single two-dimensional matrix which combines all the extracted features matrix. Here, we will get a matrix of dimension MxQ. Here M is the total number of times(action) a gesture is performed i.e., N/34 (N is described in previous sections). Q is 47 in our matrix. These 47 columns are taken as top 5 for gyroscope FFT features, 6 from gyroscope X, Y, Z Mean features for both left and right, 6 from gyroscope X, Y, Z Variance features for both left and right, 7 for EMGL DWT features, 7 for EMGR DWT features and 16 for EMG RMS features

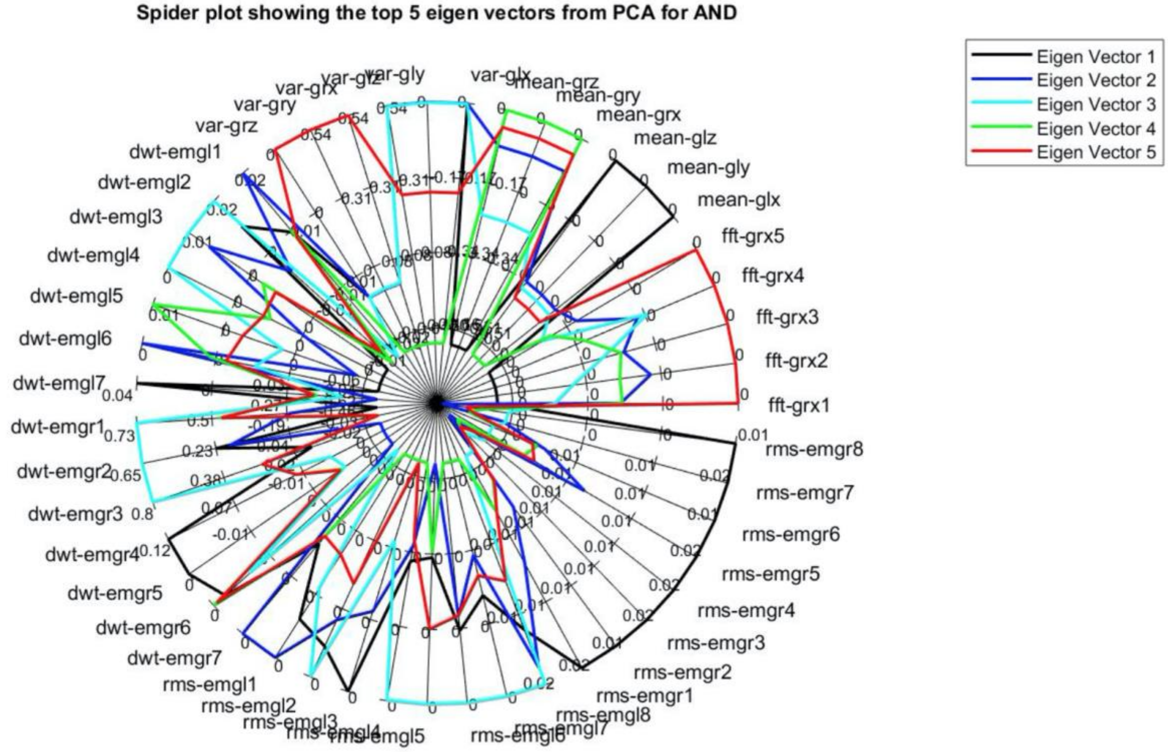
4.3.2. Execution of PCA

Here we found the covariance matrix of the data feature matrix described above using *cov* function and then we find the eigen vector matrix using the *eig* function in the MATLAB library. Below is the spider plot for the top 5 eigen vectors. The eigenvectors of the top 5 principal components are shown below as generated by *spider plot*.

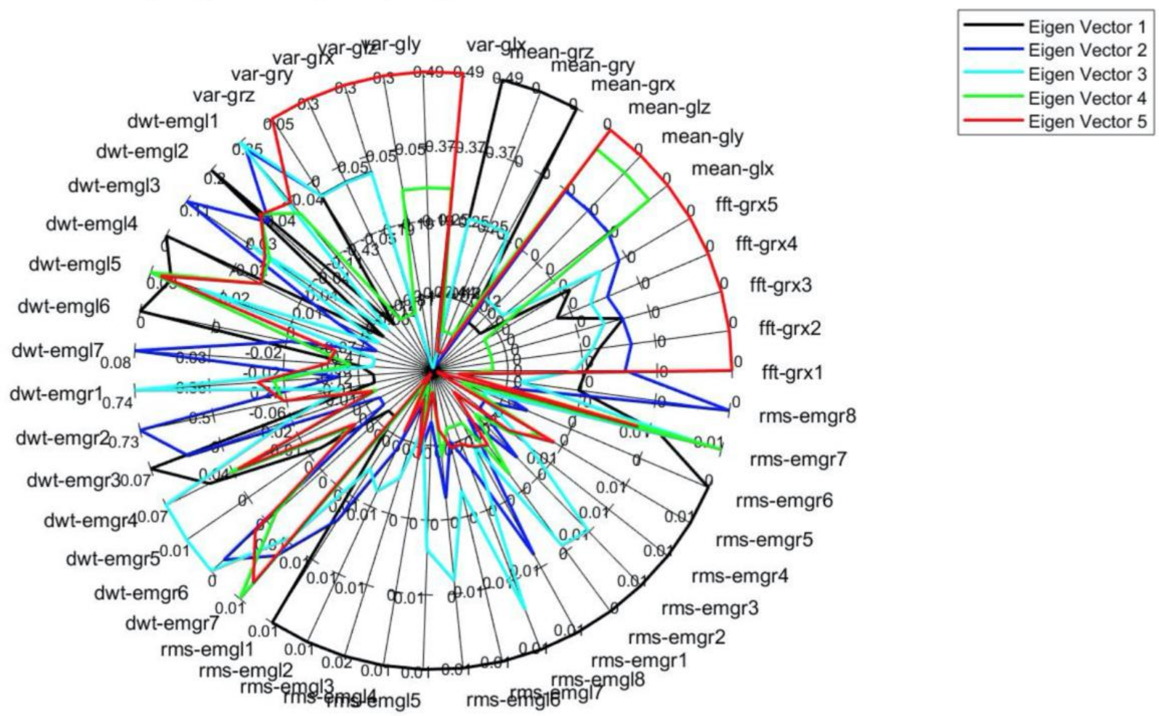
Spider plot showing the top 5 eigen vectors from PCA for ABOUT



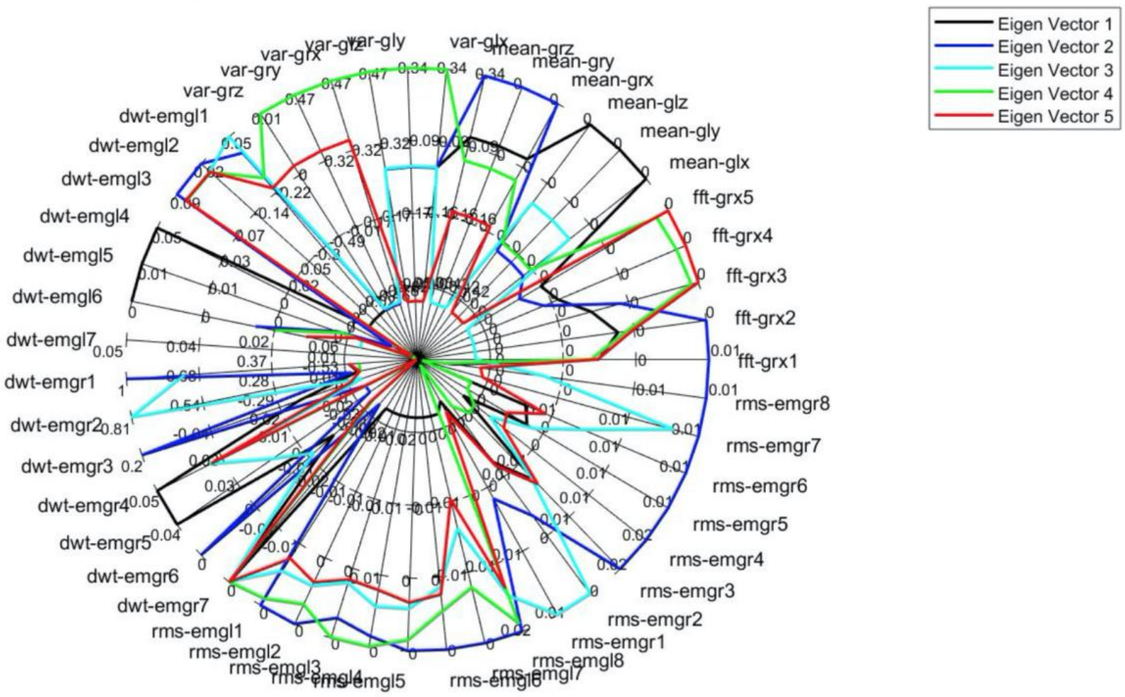
Spider plot showing the top 5 eigen vectors from PCA for AND



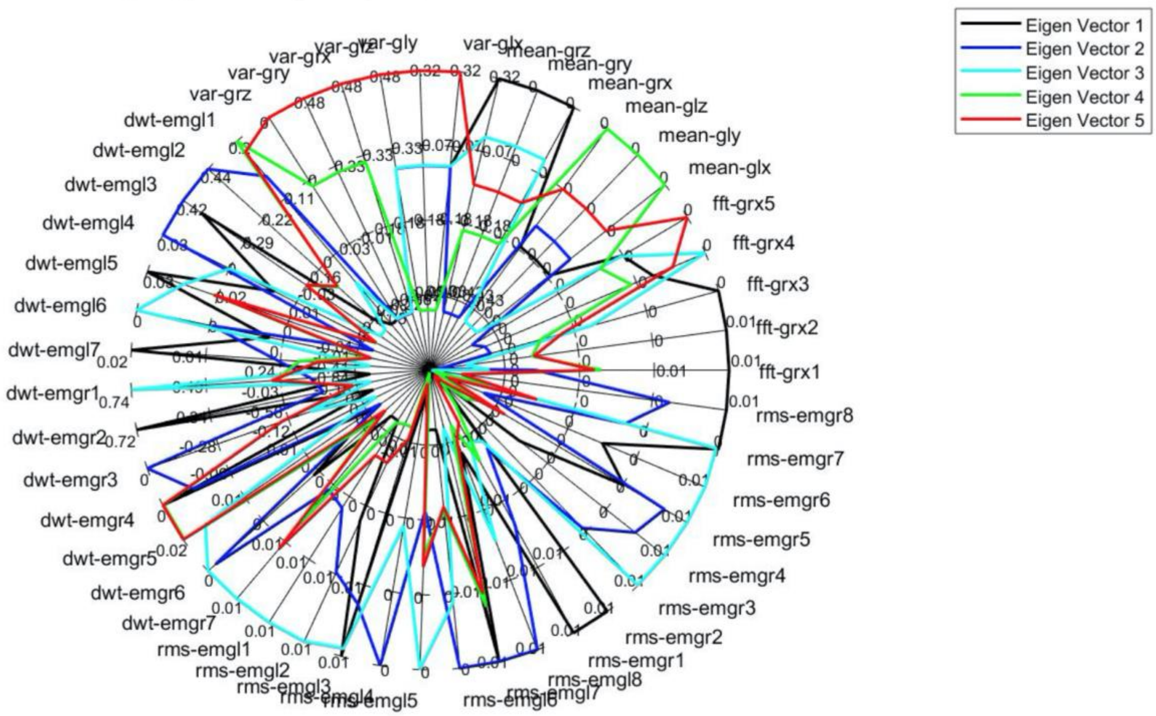
Spider plot showing the top 5 eigen vectors from PCA for CAN



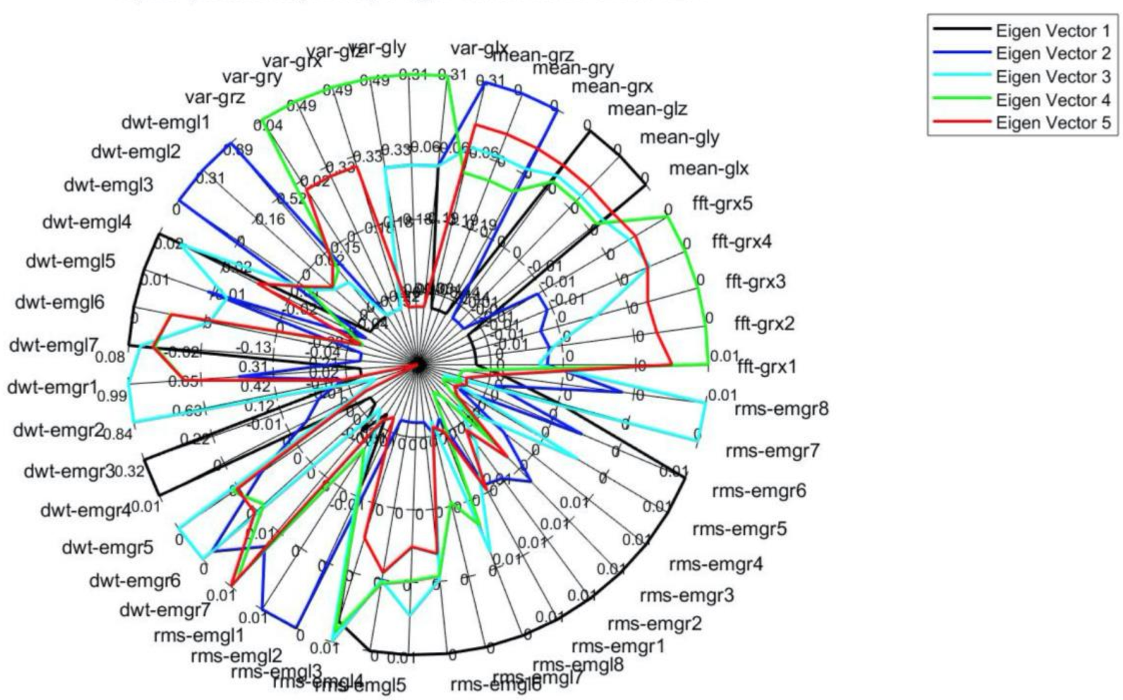
Spider plot showing the top 5 eigen vectors from PCA for COP

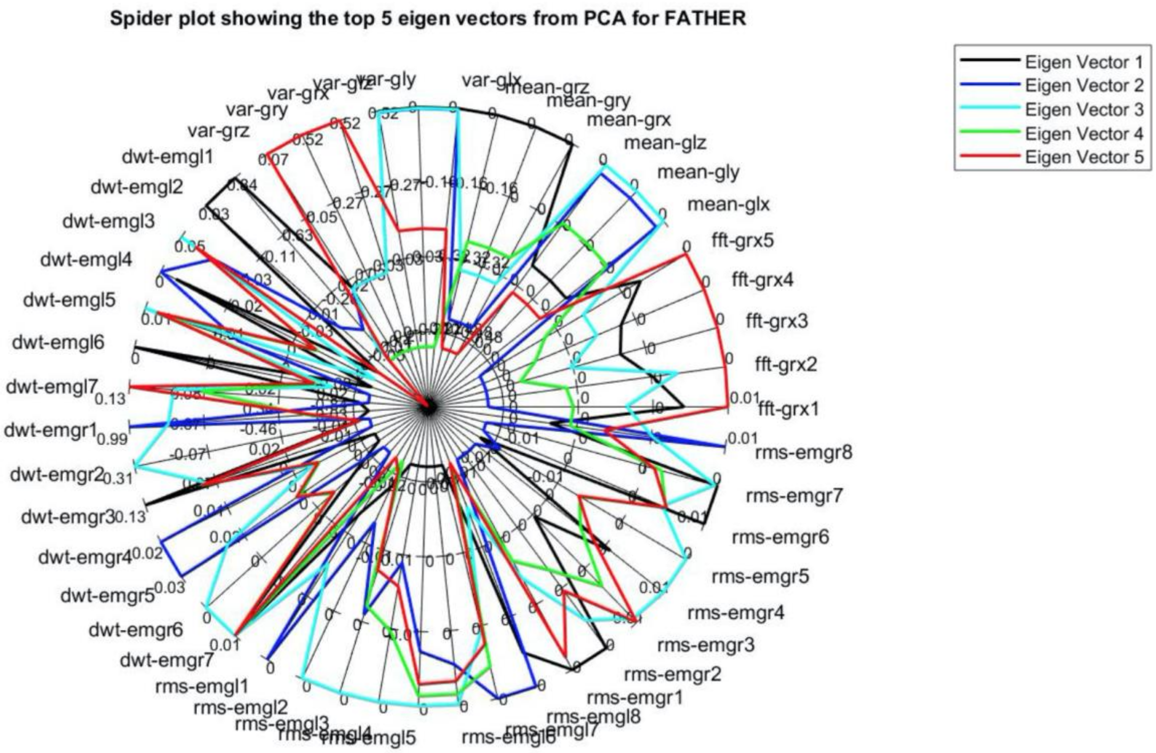


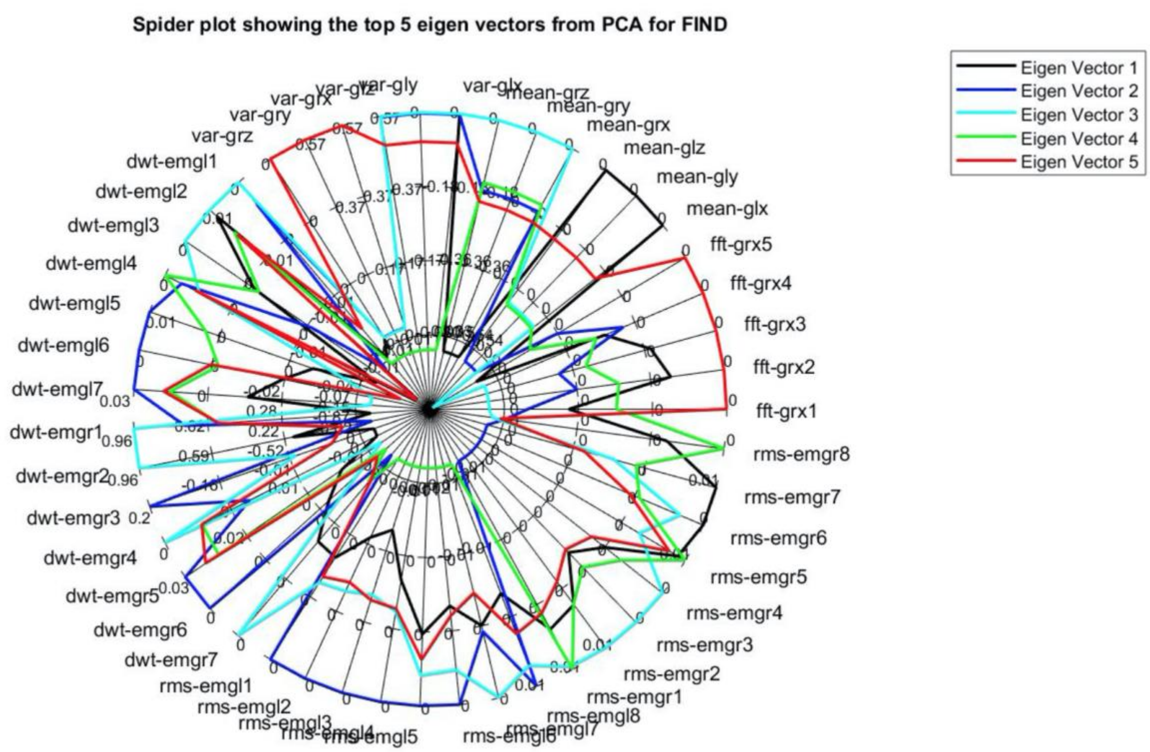
Spider plot showing the top 5 eigen vectors from PCA for DECIDE



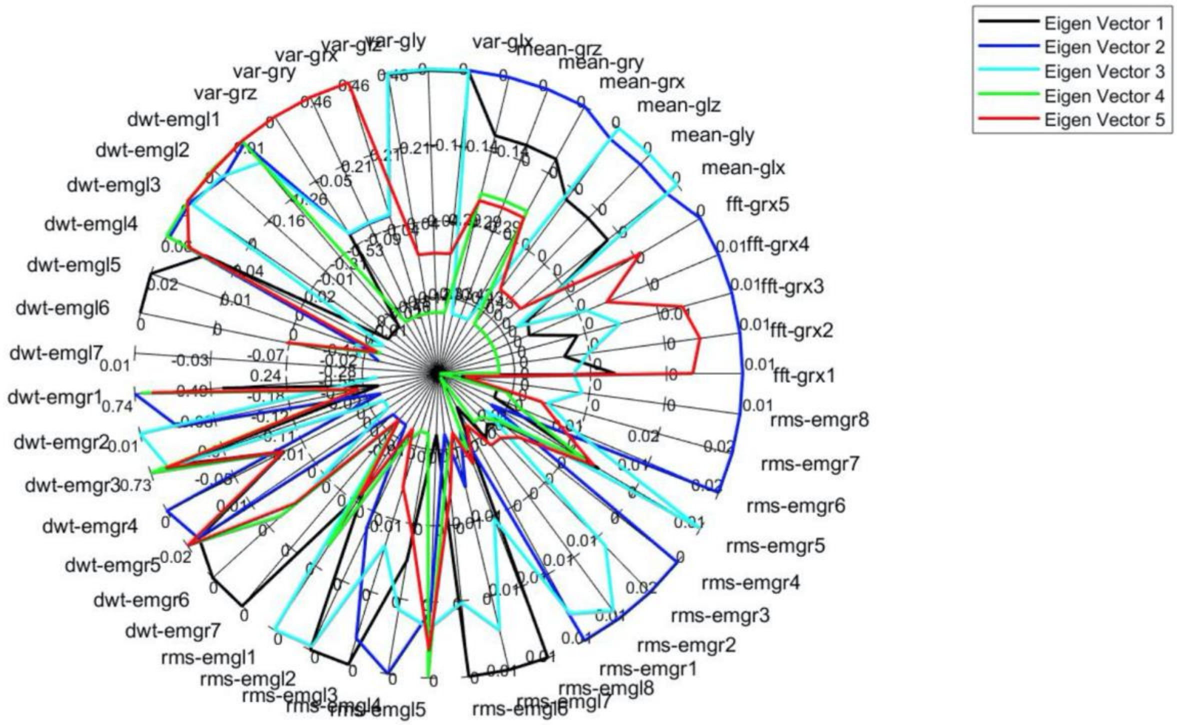
Spider plot showing the top 5 eigen vectors from PCA for DEAF



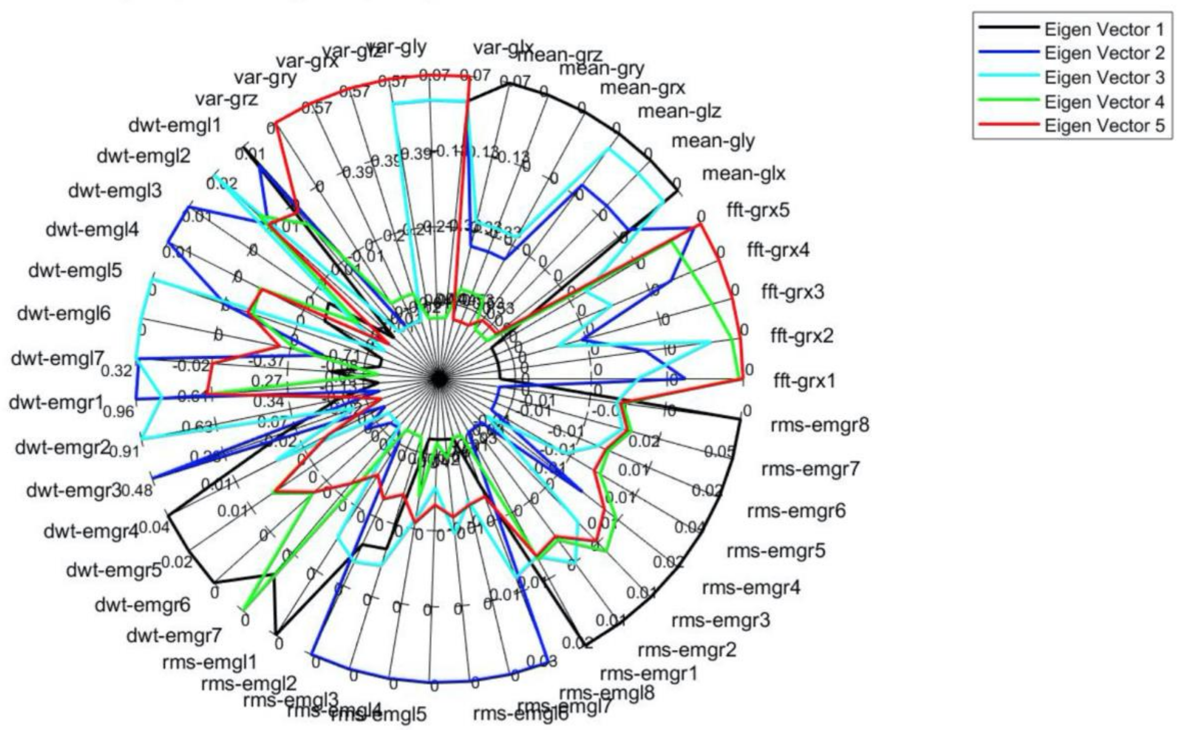

Spider plot showing the top 5 eigen vectors from PCA for FATHER
Eigen Vector 1
Eigen Vector 2
Eigen Vector 3
Eigen Vector 4
Eigen Vector 5
var-gly
var-glx
mean-grz
mean-gry
mean-grx
mean-glz
mean-gly
mean-glx
fft-grx5
fft-grx4
fft-grx3
fft-grx2
fft-grx1
rms-emgr8
rms-emgr7
rms-emgr6
rms-emgr5
rms-emgr4
rms-emgr3
rms-emgr2
rms-emgr1
rms-emgl8
rms-emgl7
rms-emgl6
rms-emgl5
rms-emgl4
rms-emgl3
rms-emgl2
rms-emgl1
dwt-emgr7
dwt-emgr6
dwt-emgr5
dwt-emgr4
dwt-emgr3
dwt-emgr2
dwt-emgr1
dwt-emgl7
dwt-emgl6
dwt-emgl5
dwt-emgl4
dwt-emgl3
dwt-emgl2
dwt-emgl1
var-grz
var-gry
var-grx
var-glz


Spider plot showing the top 5 eigen vectors from PCA for FIND
Eigen Vector 1
Eigen Vector 2
Eigen Vector 3
Eigen Vector 4
Eigen Vector 5
var-gly
var-glx
mean-grz
mean-gry
mean-grx
mean-glz
mean-gly
mean-glx
fft-grx5
fft-grx4
fft-grx3
fft-grx2
fft-grx1
rms-emgr8
rms-emgr7
rms-emgr6
rms-emgr5
rms-emgr4
rms-emgr3
rms-emgr2
rms-emgr1
rms-emgl8
rms-emgl7
rms-emgl6
rms-emgl5
rms-emgl4
rms-emgl3
rms-emgl2
rms-emgl1
dwt-emgr7
dwt-emgr6
dwt-emgr5
dwt-emgr4
dwt-emgr3
dwt-emgr2
dwt-emgr1
dwt-emgl7
dwt-emgl6
dwt-emgl5
dwt-emgl4
dwt-emgl3
dwt-emgl2
dwt-emgl1
var-grz
var-gry
var-grx
var-glz

Spider plot showing the top 5 eigen vectors from PCA for GO_OUT



Spider plot showing the top 5 eigen vectors from PCA for HEARING



4.3.3. Eigen Vectors

The variance accounted by the highest magnitude Eigen vector along with dominant components in the Eigen vector are as follows respectively:

ges_name = "AND"

78.60, rms-emgr1 through 8

ges_name = "CAN"

82.8, rms-emgl1 through 8 with rms-emgr1 through 6

ges_name = "COP"

71.03, mean-glx, mean-gly, mean-glz, dwt-emgl4 to 6

ges_name = "DEAF"

73.64, dwt-emg and rms-emgl1 - 8

ges_name ="DECIDE"

67.44, fft-grr

ges_name = "FATHER"

67.8, var-glx, mean-gry and grz

ges_name ="FIND"

84.71, mean-glx and gly and glz, rms-emgr5 to 7

ges_name = "GO_OUT"

75.76, rms-emgl6 to 8

ges_name ="HEARING"

75.05, dwt-emgr4 to 7, mean-gr and gl

ges_name = "ABOUT"

53.82, most of dwt-emgl and few dwt-emgr

4.3.4. Result of PCA

As we can see that the most of the variance has been accounted for in the first Eigen vector and we have taken the first five Eigen vectors, we can account for most of the variance from the data and different Eigen vectors of different gestures have different dominant sensors, which clearly helps in differentiating the gestures.

4.3.5. Final Conclusion

PCA was helpful in this scenario. We have reduce our feature matrix size from Nx47 to Nx5 and still capture more than 75%(average of the variance accounted by the first Eigen vector over all gestures) of the variance in the total data set. Which is helpful as features along the most variance can discriminate between classes to a good extent. If less variance along some direction then not much discrimination can occur among the classes. But if a feature does change and thus exhibits variance then it can.