Eating Activity Recognition

Kenji Mah

Ira A. Fulton Schools of Engineering, ASU Online
Arizona State University
Tempe, Arizona
kmmah@asu.edu

***Abstract*—We will determine whether someone is eating using data collected by Myo Wristbands. Machine learning algorithms such as decision trees, support vector machines, and neural networks will be used to classify the samples. We will also explore dimensionality reduction techniques, such as Principle Component Analysis, to reduce the training time needed for the classification algorithms.**

***Index Terms*—Principle Component Analysis, Feature Extraction, Feature Selection, Decision Trees, Support Vector Machine, Neural Network**

I. Introduction

To develop an application that will allow us to understand the users eating actions we must first organize how we are going to approach this problem. Typically, machine learning workflows in summary start off with preparing the data, applying algorithms to extract knowledge about the data, and then go through an iterative process of trial-and-error to improve upon the results [1]. The data comes from video footage and Myo wristbands that contain inertial measurement unit (IMU) sensors and electromyography (EMG) sensors. We are going to use supervised learning techniques because we are given timestamps of the video data where is binarily labeled with the strings 'eating' and 'non-eating'. The steps we will go through will be outlined by the following activities: Data cleaning and organization, feature extraction, feature selection, test and train split, machine learning models, analysis of the results.

II. Data Cleaning and Organization

Data format

We are provided text files that contain the raw values at given Unix timestamps, however we do not have the raw video data and instead we are provided with the time spans of eating actions that correspond to the frames of the video data and is what we will be calling the "ground truth" and is what we will be using to label the IMU and EMG data. In order to synchronize the different framerates between the Myo sensors and the video data, we must multiply the ground truth values with the Myo sensor data and divide it by the video frame rate. The IMU, EMG, and video data are gathered at a frequency of about 50, 100, and 30 Hz respectively. We were told that the end time for the Myo and Video timestamps is more accurate than using the start time. Therefore, we can synchronize two datasets by setting the last frame time to the last UNIX time stamp in IMU or EMG file and progressively label them from finish to start. We also assume that each row in the Myo data corresponds to a single frame. This assumption makes it more difficult to align the EMG and IMU data because of the slight inconsistencies with the rate at which the IMU and EMG sensors capture. We decided to continue to use only the IMU data for the rest of the project because of this complication.

The IMU data contains the following features: UNIX time stamp, Orientation X, Orientation Y, Orientation Z, Orientation W, Accelerometer X, Accelerometer Y, Accelerometer Z, Gyroscope X, Gyroscope Y, and Gyroscope Z.

Data Anomalies

There are a few anomalies that we have found with the raw data. The first is that we must add a comma in the file 1503609551913.txt, from user19/spoon to fix the missing byte of data. There is also an insufficient amount of data regarding the raw Myo data for user 18 fork data and user 25 spoon data. If we calculate the minimum number of seconds that the ground truth spans, it does cover the amount of time that the Myo data covers. In fact, the time span of the Myo data was significantly less than their respective ground truth files. Because it is not clear how this mistake occurred it would be best to eliminate those specific parts and just consider them as noise.

III. Feature Extraction

Feature extraction is important for this situation because of the scope of our data. Every frame represents about $1/100$ of a second. In context, we do not classify if a person is eating or not eating for such at every given millisecond. At least a second or two is needed to classify eating actions. The aggregation of data and extracting features from them will be more useful in this case and may also provide more insight than the original features.

After the synchronization and labialization of the data, we are now left with a time sensitive dataset with 11 features and a bunch of data for an individual moment in time of an eating action. In order to get a more representative data point that better captures an eating action, we will aggregate multiple rows and extract certain features from them. From each of the original 11 features we will aggregate every consecutive eating frame and use a sliding window of 100 frames with a stride

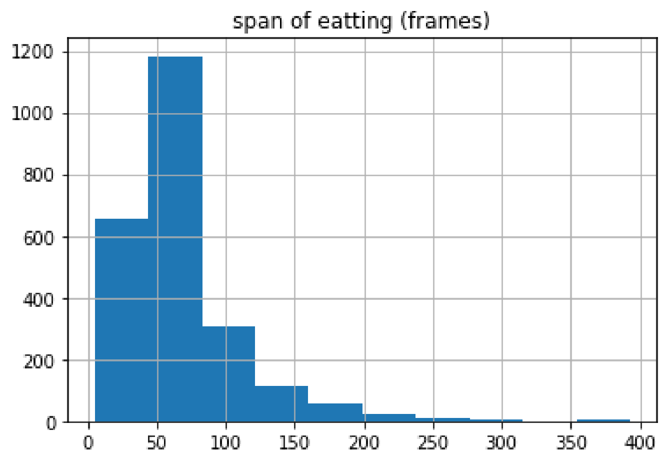


Fig. 1. IMU distribution of eating frame lengths [x-axis: span of frames, y-axis: occurrences]

of 100. We will then do the same with consecutive non eating frames so that we do not mix up the eating and non-eating samples. The reason why we are using rows of 100 is because when we explored the distribution as shown in Figure (1) of the span of the eating frames, it seemed like a reasonable window size. It is important to note that we are keeping the timestamps in order in order to preserve the potential time sensitive information from these samples. From these windows we will be extracting the following features from the raw values that the Myo wristbands output: Fast Fourier Transformation, Average, Standard Deviation, Maximum, and Minimum.

The Fast Fourier Transformation (FFT) transforms a signal from its temporal domain, to a representation in frequency domain and is useful for signal processing [2]. This means that if there are any sequential patterns in acceleration and deceleration, orientation, or other attributes, FFT will approximate the frequencies that represent those patterns. We will use the Numpy library to perform the FFT. By capturing the top 5 most popular frequencies from the frequency spectrum we avoid the problem of capturing the 0 Hz frequency that typically shows up as the most popular and we also avoid potential noise frequencies that the sensors capture.

Extracting the average attempts to capture helpful information about potential differences in eating and non-eating actions, whereas the standard deviation will capture different variations between the action. Maximum and minimum values may be useful if a certain feature reaches a critical value when a certain action is performed. Our intuition for extracting the features, average, STD, max, and min, are aimed more towards distinguishing activity and non-activity, whereas FFT is aimed more towards distinguishing the different actions. These feature extractions increase the number of attributes of the data from 11 to 90. We will have some features that do not show much variation such as the 0 Hz FFT frequency and we will eliminate such features in the feature selection step of our project.

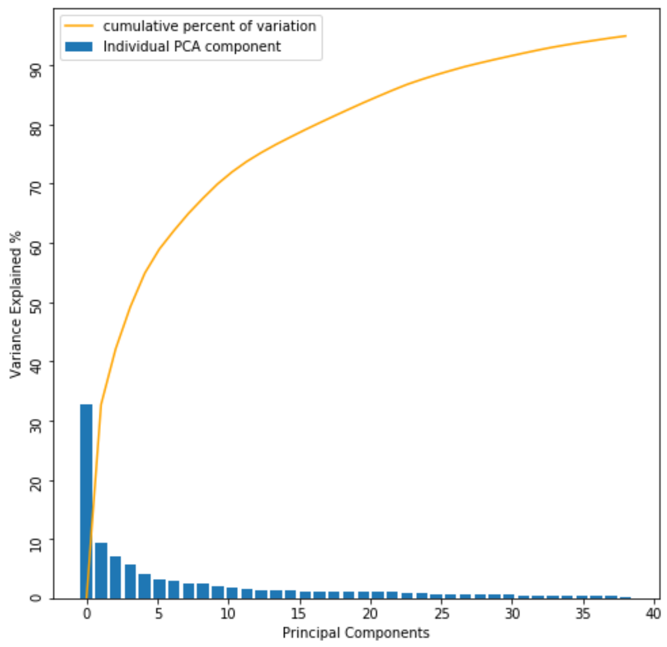


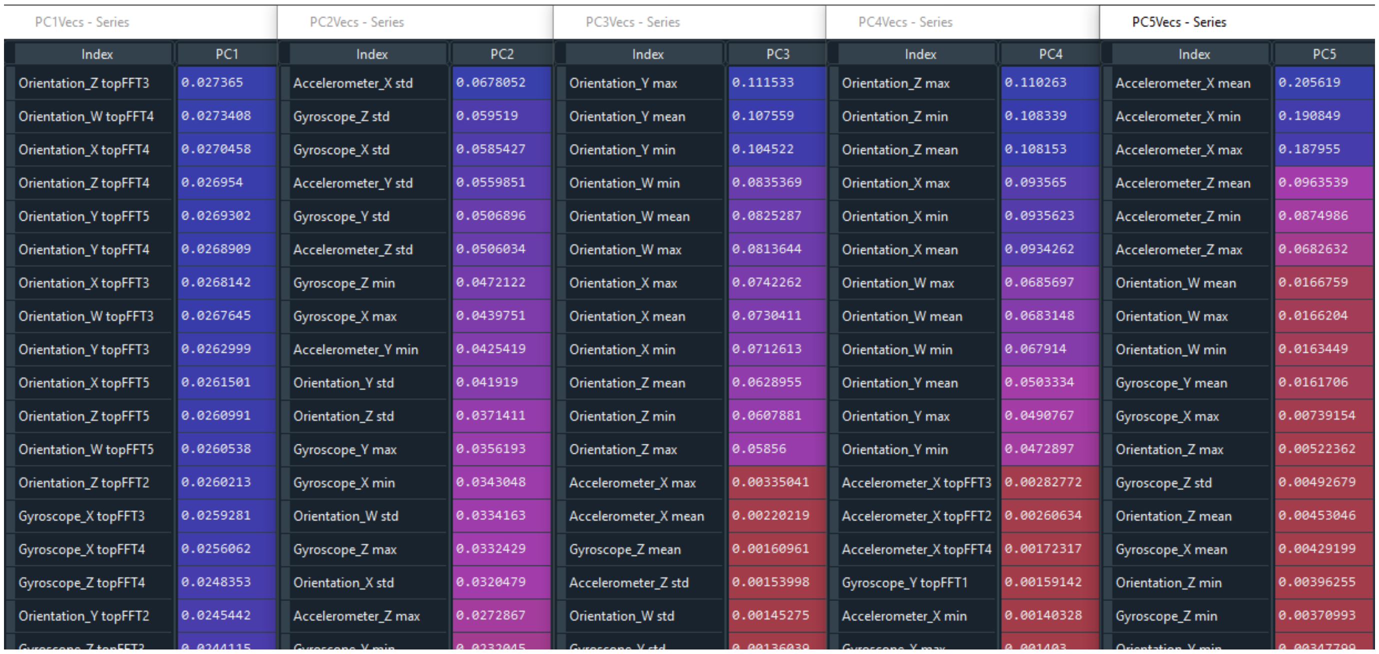
Fig. 2. Scree plot of first 38 Components when 95% of variance is kept

IV. Feature Selection

Feature selection is important for speeding up our machine learning algorithm by eliminating irrelevant features in the dataset. We will be using Principle Component Analysis (PCA) as a dimensionality reduction technique. PCA in short applies an orthogonal transformation to the data into principle components such that the first principle has the largest variance while the last component has the least variance. We used Sklearns library to perform a PCA where the minimum number of components is kept in order to retain 95% of the total variance in the data. After doing this, the resulting number of components reduces from 90 to 38. By calculating the explained variance of each principle component, we can get an understanding how much variance each principle component contributes. From this we get the resulting scree plot as shown in Figure (2). These values use the eigenvalues of the resulting matrix which, as stated by UCLA Institute for Digital Research and Education Statistical Consulting [3], "represents the total amount of variance that can be explained by a given principal component."

By further exploring the Principle Components we can start to accumulate a general understanding of the amount of variance that each attribute contributes to the dataset. Figure (3) shows a portion of the square of the eigenvectors for the first 5 Principle Components. The vectors are rearranged in descending order to show the most impactful attributes that contribute to that component.

An analysis of each principle component shows that we might have promising results from the FFT features because they are the top features accounting for the first Principle Component's variance, however we cannot conclude that they



PC1Vecs - Series		PC2Vecs - Series		PC3Vecs - Series		PC4Vecs - Series		PC5Vecs - Series	
Index	PC1	Index	PC2	Index	PC3	Index	PC4	Index	PC5
Orientation_Z topFFT3	0.027365	Accelerometer_X std	0.0678052	Orientation_Y max	0.111533	Orientation_Z max	0.110263	Accelerometer_X mean	0.205619
Orientation_W topFFT4	0.0273408	Gyroscope_Z std	0.059519	Orientation_Y mean	0.107559	Orientation_Z min	0.108339	Accelerometer_X min	0.190849
Orientation_X topFFT4	0.0270458	Gyroscope_X std	0.0585427	Orientation_Y min	0.104522	Orientation_Z mean	0.108153	Accelerometer_X max	0.187955
Orientation_Z topFFT4	0.026954	Accelerometer_Y std	0.0559851	Orientation_W min	0.0835369	Orientation_X max	0.093565	Accelerometer_Z mean	0.0963539
Orientation_Y topFFT5	0.0269302	Gyroscope_Y std	0.0506896	Orientation_W mean	0.0825287	Orientation_X min	0.0935623	Accelerometer_Z min	0.0874986
Orientation_Y topFFT4	0.0268909	Accelerometer_Z std	0.0506034	Orientation_W max	0.0813644	Orientation_X mean	0.0934262	Accelerometer_Z max	0.0682632
Orientation_X topFFT3	0.0268142	Gyroscope_Z min	0.0472122	Orientation_X max	0.0742262	Orientation_W max	0.0685697	Orientation_W mean	0.0166759
Orientation_W topFFT3	0.0267645	Gyroscope_X max	0.0439751	Orientation_X mean	0.0730411	Orientation_W mean	0.0683148	Orientation_W max	0.0166204
Orientation_Y topFFT3	0.0262999	Accelerometer_Y min	0.0425419	Orientation_X min	0.0712613	Orientation_W min	0.067914	Orientation_W min	0.0163449
Orientation_X topFFT5	0.0261501	Orientation_Y std	0.041919	Orientation_Z mean	0.0628955	Orientation_Y mean	0.0503334	Gyroscope_Y mean	0.0161706
Orientation_Z topFFT5	0.0260991	Orientation_Z std	0.0371411	Orientation_Z min	0.0607881	Orientation_Y max	0.0490767	Gyroscope_X max	0.00739154
Orientation_W topFFT5	0.0260538	Gyroscope_Y max	0.0356193	Orientation_Z max	0.05856	Orientation_Y min	0.0472897	Orientation_Z max	0.00522362
Orientation_Z topFFT2	0.0260213	Gyroscope_X min	0.0343048	Accelerometer_X max	0.00335041	Accelerometer_X topFFT3	0.00282772	Gyroscope_Z std	0.00492679
Gyroscope_X topFFT3	0.0259281	Orientation_W std	0.0334163	Accelerometer_X mean	0.00220219	Accelerometer_X topFFT2	0.00260634	Orientation_Z mean	0.00453046
Gyroscope_X topFFT4	0.0256062	Gyroscope_Z max	0.0332429	Gyroscope_Z mean	0.00160961	Accelerometer_X topFFT4	0.00172317	Gyroscope_X mean	0.00429199
Gyroscope_Z topFFT4	0.0248353	Orientation_X std	0.0320479	Accelerometer_Z std	0.00153998	Gyroscope_Y topFFT1	0.00159142	Orientation_Z min	0.00396255
Orientation_Y topFFT2	0.0245442	Accelerometer_Z max	0.0272867	Orientation_W std	0.00145275	Accelerometer_X min	0.00140328	Gyroscope_Z min	0.00370993
Gyroscope_Z topFFT3	0.0244115	Gyroscope_Y min	0.0232045	Gyroscope_Y std	0.00136039	Gyroscope_X max	0.001403	Orientation_Y min	0.00347799

Fig. 3. First 5 Principle Component Eigenvectors rearranged by descending value

will provide the most insight during classification. This is because PCA is an unsupervised technique aimed towards extracting the top variance of the data. We also gain some confidence in our feature extractions by analyzing the PCA eigenvectors. One example of this is that the lowest FFT that we predicted would contain the 0 Hz frequency shows up as the lowest values in the eigenvector.

These eigenvectors will then be used to multiply the original feature matrix in order get the new reduced dimensional matrix that will be used for reducing the time it takes to train our machine learning algorithms. This reduction only reduced the number of features and not the samples of the data, therefore our PCA reduced our feature matrix by a little more than a factor of two.

V. Train and Testing Split

Different methodologies for training and testing splits will produce different results and can provide insight on the data in general. In our case we will be observing the differences of a user dependent split and a user independent split.

For the user dependent split, we will first group up all the eating and non-eating by user. Because there is a significant imbalance in the number of eating and non-eating samples, we will perform down sampling on each user so that the number of eating and non-eating instances match. For each user we will divide the data where 60% will be the training set and 40% will be the test set. All the training data is then combined to form a single matrix for the training set, and all the test data will be combined to form a single test set.

For the user independent split, we will randomly choose 18 out of 30 (60%) of the users to be in the training data and the rest in the test data. We will then perform down sampling to balance the classes. This case differs from the user dependent split because a user is either in the training set or in the test set. This methodology is predicted to perform worse than the user dependent split because an individual's pattern of eating movements may differ from another individuals.

VI. Machine Learning

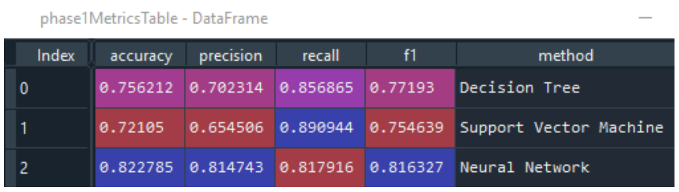
We will use and compare the results of three different machine learning algorithms each with two different training and testing split methodologies. Because we are trying to identify the differences between the two split methods, all the parameters for the machine learning algorithms will be consistent for both methods. For all the machine learning models we will be using implementations from the Sklearn library.

A. *Decision Trees*

For our decision tree we will be using the Gini impurity measure for measuring the quality of our splits. Because the Sklearn library does not have the functionality for post pruning the tree, we used a pre pruning method by setting a max limit of the depth of the tree to 6. This algorithm uses the greedy strategy when iteratively choosing the split.

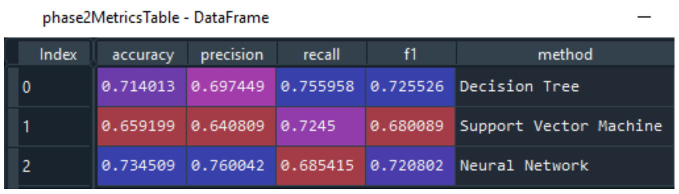
B. *Support Vector Machine*

For our support vector machine, we will be using a linear kernel with a tolerance of 1 x 10^{-3}. We decided not to set a limit on the number of iterations therefore the SVM will find the best fitting hyperplane with our given tolerance for the data.


phase1MetricsTable - DataFrame

Index	accuracy	precision	recall	f1	method
0	0.756212	0.702314	0.856865	0.77193	Decision Tree
1	0.72105	0.654506	0.890944	0.754639	Support Vector Machine
2	0.822785	0.814743	0.817916	0.816327	Neural Network

Fig. 4. Results of the user dependent split


phase2MetricsTable - DataFrame

Index	accuracy	precision	recall	f1	method
0	0.714013	0.697449	0.755958	0.725526	Decision Tree
1	0.659199	0.640809	0.7245	0.680089	Support Vector Machine
2	0.734509	0.760042	0.685415	0.720802	Neural Network

Fig. 5. Results of the user independent split

C. Neural Network

For our neural network we will use the Limited-memory Broyden–Fletcher–Goldfarb–Shanno algorithm (lbfgs) as our optimizer, a learning rate of 1 x 10^{-5}, hidden layers size of (38, 30). We set the seed for the initial random state to 1 for replication purposes and set a max limit of iterations to 20,000 just in case the algorithm does not converge.

VII. Results

The evaluation of the machine learning algorithms will be measured on the test data on their accuracy, precision, recall, and F1 score. The user dependent split's results are displayed in Figure (4) and the user independent in Figure (5). The user dependent split performs significantly better than the user independent split. These results support our hypothesis that users may have different eating habits. The fact that both splits show at least 70% on most metrics proves that there is a general distinction between eating and non-eating actions. It seems that the neural network performs significantly better than the decision tree and the SVM, whereas the decision tree performs slightly better than the linear SVM, but not by much.

VIII. Lessons Learned

This project taught me how to approach treating an unprocessed dataset and use algorithms to make it useful. Techniques for visualizing, preprocessing, and applying the data were practiced so that the next full stack machine learning project can become more fluid and more through.

IX. Conclusion

In conclusion, we were successful in creating a classification algorithm for eating and non-eating actions using data from the Myo wristbands. Techniques for the extraction data from time sensitive information, reduction in the dimensionality of the data with the use of PCA, and application of machine learning algorithms all proves to be useful for this classification task. However, there are still improvements that could be made to significantly improve our results.

The improvements such as attaining the missing IMU and EMG data as stated in the data anomalies section will not be mentioned in this list because we do not have control over that aspect of the project. We can however, improve our assumption that every row represents a single frame. Instead we could convert the ground truth data into the time domain and use the Unix timestamps instead. This however is slightly harder and more time consuming to implement and may be inaccurate due to rounding errors, but it could change the labeling of the data slightly. This approach may however allow us to concatenate the IMU and EMG data. Previous attempts to concatenate these two different datasets proved difficult because the differences in the number of samples after the feature extraction step between the IMU and EMG data was too great. The misalignment of the data hinders results of this project so that is why we only worked with the IMU data.

The aggregation window can provide significant changes depending on the window size and stride of the window. I believe that the window size is adequate however I believe there would be significant changes if we changed the stride of the window to a smaller value. This would significantly increase the time it would take to do feature extraction because you must do multiple passes over the data. This would be more useful for a real time recognition system thought because the data is constantly being uploaded.

The machine learning algorithms can also be improved to a more complex model. The Neural Network performed the best out of the other models since of the complexity of its complexity. Methods such as post pruning the decision tree or changing the kernel of the SVM to be polynomial kernel may provide the extra complexity needed to see improvements in the results. We could also alter the complexity of the neural network by using different hidden layer dimensions. For all machine learning models, an implementation of Receiver operating characteristic (ROC) curves with various parameters for each model would be helpful to determine which parameters produces the most optimal results.

There are many variations of this project but attempting them all will be an impossible task. Even though there are no perfect machine learning algorithms, there are useful ones.

References

[1] L.Ma, A.Parameswaran, S.Song, D.Xin, ”How Developers Iterate on Machine Learning Workflows,” University of Illinois, Urbana-Champaign, 2018. Available: https://arxiv.org/abs/1803.10311. [Accessed March 15, 2021]

[2] D. Rockmore, ”The FFT - an algorithm the whole family can use,” Departments of Mathematics and Computer Science, Dartmouth College, Hanover, NH, October 11, 1999

[3] “PRINCIPAL COMPONENTS (PCA) AND EXPLORATORY FACTOR ANALYSIS (EFA) WITH SPSS,” UCLA Institute for Digital Research and Education Statistical Consulting. [Online]. Available: https://stats.idre.ucla.edu/spss/seminars/efa-spss/. [Accessed: 29-Feb-2020].