Using Machine Learning Techniques to Differentiate Between Eating and Other Activity in Electromyography Data

Cale Dunlap

***Abstract*—A collection of Electromyography [1] data from the dominate arms of thirty users is classified as "eating" or "not eating". This data is reduced in dimensionality, analyzed for principal components, and used to train four different machine learning algorithms that can predict the classification of test data that omits the classification label. The four machines are trained to predict eat actions from a user-dependent and user-independent perspective and their performance is compared across precision, recall, and F1 scores.**

I. Introduction

The goal of this project is to use machine learning techniques to interpret Electromyography data from thirty users and train a series of algorithms to predict when that person is eating or not. The raw input data provided was in the form of sparse CSV files totaling over 1.7 million records with eight EMG features and a timestamp in milliseconds since the UNIX epoch. I was also provided ground truth data in the form of a matrix that contains start and end frames from a video that identifies at which point in the EMG data a user is eating or not. The classification for each EMG record is derived from this ground truth data.

I was required to clean and aggregate the data, analyze the data to select meaningful features, perform principal component analysis to transform it into a reduced feature space and finally train three machine learning algorithms: a decision tree, a support vector machine, and a multi-layer neural network. I chose to train a second SVM but with a different kernel, bringing the total machine learning systems to four. The training and testing was split into two parts: user-dependent and user-indepenent methods. The user-dependent method is training the set of machines within the scope of each individual user and the user-independent method is training a single set of machines for all users. After training and testing the results are compared by several well-known machine learning performance metrics: precision, recall, and F1 score [2].

II. Explanation of the Solution

In order to derive ground truth information from the EMG data I had to pair it with the ground truth frames from a video in order to classify each row. I was not provided the video itself, so I was given start and end frames of the video that indicated when a person is eating. Along with these ground truth frames I was provided metadata about the video: length in frames per second, and its frame rate. I was instructed that an assumption I could take is that the end of the video and the end of the EMG sensor data was close to the same, closer than the beginnings. So given the last timestamp in the EMG data (τ), the frame rate of the video (ρ), the length of the video in frames per second (ω) and the eating frames ($F_{start}$and F_{end}), the start and end timestamps in the EMG data for eat classification can be calculated with the following formulas: $T_{start} = \frac{\tau-\omega-F_{start}}{1000\rho}$ and $T_{end} = \frac{\tau-\omega-F_{end}}{1000\rho}$respectively. I marked all frames in the EMG data between these two timestamps with a 1 in the 'eat' column. All others are marked with a 0. A sample of this data is shown in Table I with an abbreviated timestamp.

TABLE I
EXAMPLE EMG DATA WITH CLASS LABEL 'EAT'

ts	emg1	emg2	emg3	emg4	emg5	emg6	emg7	emg8	eat
49719	-1	-2	-1	-3	-2	-1	2	0	0
49726	0	0	0	-2	0	-1	0	0	0
49741	-3	0	0	-1	-2	-1	-1	-1	1
49749	0	1	-1	-1	-3	-2	-1	0	0
49756	0	-3	-3	0	0	0	-4	-2	1

Once the data has been classified, it is separated into four different matrices prior to being analyzed for meaningful features: eating with a fork, not eating with a fork, eating with a spoon, and not eating with a spoon. Then a 2000 record sample is extracted at random from each matrix and then row-wise statistical aggregation is performed in batches of 200 rows across each EMG feature. These statistical functions are: minimum, maximum, mean, standard deviation, and root-mean-square. Plots are created for each statistic for each feature and their eat and non-eat result within each of the two utensils is compared. An example set of plots of the aggregated minimums for one user's fork actions is shown in Figure 1.

Visual identification of meaningful features is performed by subjectively reviewing each plot for noticeable differences between eat and non eat series. In Figure 1 for example, EMG1's minimum measurements differ in range between eating and non-eating labels. Therefore, this feature as well as 6 other features are selected from the original union of all user and utensil data for principal component analysis: minimums of EMG 4 & 8, mean of EMG 4, and standard deviations of EMG 1, 7, & 8. A sample of the feature reduced matrix is shown in Table II. The new feature space is 58,000 rows and 7 feature columns plus 1 class column.

Cale Dunlap (e-mail: cjdunla1@asu.edu)

TABLE II
NEW INPUT FEATURE SPACE PRIOR TO PRINCIPAL COMPONENT ANALYSIS

emg_1_min	emg_4_min	emg_8_min	emg_4_mean	emg_1_std	emg_7_std	emg_8_std	eat	user
-19	-25	-20	4.571429	19.423844	4.680252	8.118175	0	user09
-32	-47	-28	-8.857143	13.952300	6.876461	15.510365	0	user09
-19	-11	-7	-0.428571	15.892496	4.685337	10.132456	0	user09
-44	-7	-51	2.142857	29.824087	9.378293	22.007574	0	user09
-62	-9	-53	-2.285714	27.907159	11.385036	45.814325	0	user09

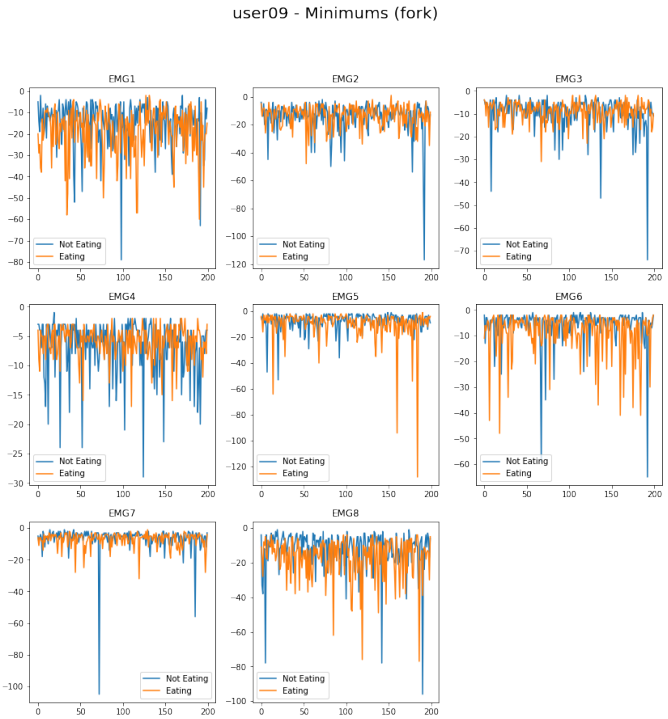

Fig. 1. Sample set of plots for minimum values across all EMG features for a single user's fork data

For the user-dependent system, principal component analysis is performed on each user's data prior to training the machines. The result of principal component analysis is a set of eigenvectors that explain the most variance in the data with respect to the class in a dimensional space equal to that of the input features [3], [4]. In this case, 5 eigenvectors are selected, which explains a total of 98.7% of the variation between the class labels for user09. These eigenvectors can be visualized across the original features using a radar plot as shown in Figure 2. In the user-independent system, principal component analysis is performed only once across a sample of every user's data to build a more generic model.

These eigenvectors are multiplied with the original feature set to create a new, reduced, feature space that is used to train the machine learning algorithms. The resulting transformation has no meaningful feature labels as these new features are projected into a different space across the data matrix, that is, each column is a blended selection of the original features meaningful to that specific eigenvector [5] . This new feature space is shown in Table III.

TABLE III
RESULTING TRANSFORM OF THE EIGENVECTORS WITH THE ORIGINAL FEATURE MATRIX. THE NEW FEATURE SPACE.

0	1	2	3	4	eat
-8.985145	-4.410992	1.487297	2.771683	-1.488867	1
-1.692692	-2.440329	-3.597773	3.536166	-1.687581	0
-7.348602	9.686857	-3.748312	-0.587658	0.935341	0
-7.296082	-1.611847	6.445079	3.018755	-1.840824	1
-9.323671	-5.188330	3.396582	-3.925569	0.627399	1

From this new feature space input, training and test data is

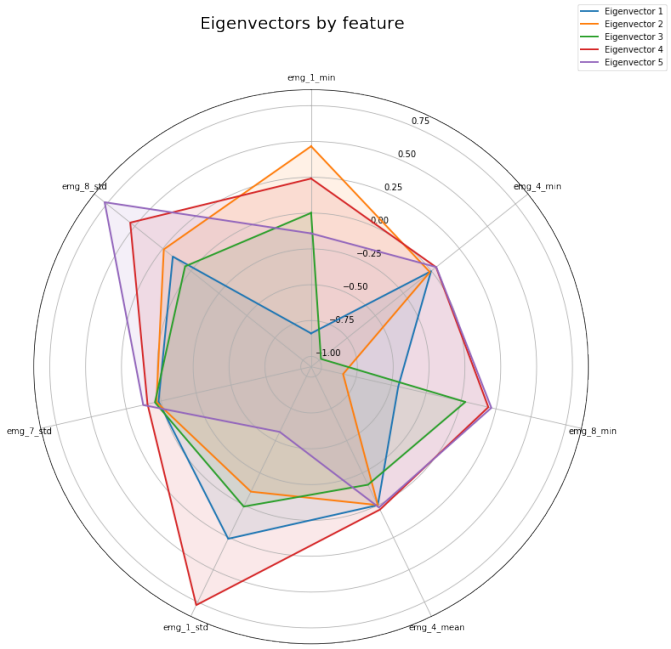

Fig. 2. Principal component analysis results visualized as a radar plot.

extracted. For the user-dependent systems, each user's data is divided such that 60% of it is used for training and the remaining 40% is used for testing. In the user-independent system, this division is done across the users such that 60% of the users are assigned to train the algorithms and the remaining 40% are assigned to test. This training and test data is used to train and test four machine learning algorithms. The first algorithm is a decision tree using a best-split algorithm, choosing the split quality by utilizing the Gini impurity [6]. The second and third type of algorithms are support vector machines with linear and radial basis function kernels, respectively. The fourth type of algorithm is a multi-layer neural network with three hidden layers with five nodes in each layer with a rectified linear unit [7], activation, a stochastic gradient-based [8] optimizer, and a constant learning rate of 0.001.

III. Description of the Results

The results from the user-dependent systems are averaged and compared to the single, generic, user-independent system's performance. The metrics used to compare performance are precision, recall, and the F1 score [2]. The average precision for the user-dependent machines is 82.8%, average recall is 84.1%, and the average F1 score is 83.3%. The precision, recall, and F1 scores for the user-independent machines is 77%, 83.9%, and 80.2% respectively. The neural network performed the best across all three scores in both user-dependent and user-independent use cases. The only exception was that the SVM with RBF kernel performed better only by a small margin in the user-independent use case's recall score. The results for each machine are detailed in Table IV and Table V and a graphical comparison is seen in Figure 3.

As could be expected, the user-independent systems performed better at predicting a user's eat action versus the user-independent system. The user-dependent systems were trained

TABLE IV
Mean User-Dependent Machine Performance

	Decision Tree	Linear SVM	RBF SVM	MLNN
Precision	0.815015	0.780929	0.852798	0.864722
Recall	0.807714	0.798621	0.874250	0.886707
F1	0.811028	0.786044	0.862343	0.875100

TABLE V
User-Independent Machine Performance

	Decision Tree	Linear SVM	RBF SVM	MLNN
Precision	0.758607	0.726007	0.792106	0.804280
Recall	0.791417	0.695333	0.938250	0.930083
F1	0.774665	0.710339	0.859007	0.862619

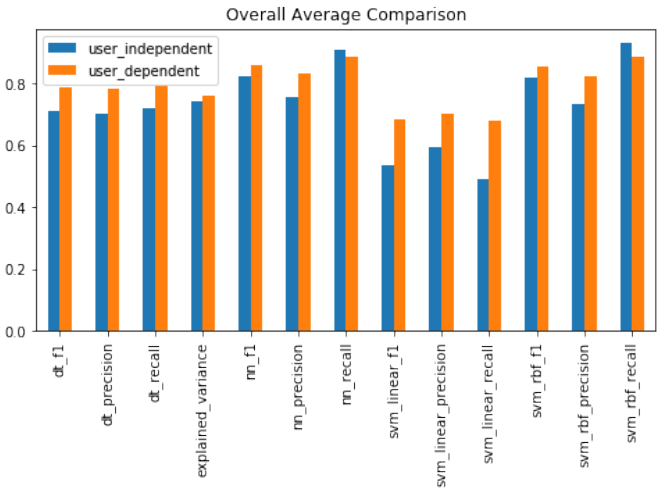

Fig. 3. Comparison of metrics between user-dependent and user-independent use cases

to recognize one single user's eating actions and therefore fit each user's data better versus the more generic user-independent system that had to fit a broader range of training data.

IV. My Contribution to the Project

This project was a solo project assigned during CSE572: Data Mining. There were no team members. However, I would like to thank the other students in my class that provided open, helpful, discussion with each other so that we could all do our best work.

V. Explanation of New Skills and Knowledge

This project taught me the importance of dimensionality reduction through aggregation and principal component analysis. Many algorithms suffer from the curse of dimensionality [9], [10] and can break down quickly when the data sets are too large. Machine learning algorithms may over-fit the data and only work in very specific cases. Dimensionality reduction though principal component analysis reduces those chances and simplifies the training model [3]. So following Occam's Razor principal, the simper models are preferred if their outcomes are effectively the same as their more complex counter-parts [11].

The different use cases in this project, user-dependent and independent also taught me how to approach machine learning models depending on their use cases. For example if I am ever going to build a machine learning system for a wide range of users, I may choose a user-specific training system depending on the level of accuracy I need and computing power I have available. If I only need moderate accuracy or don't have a lot of computing power, I may choose a system that is user-independent instead.

From a more general perspective of learning new programming languages and libraries, this project taught me how to interact with large datasets using Python, Pandas, and NumPy as well as create machine learned classifiers using Sci-Kit. These skills will be important throughout my career as I am currently employed at a company that specializes in extracting meaningful information from occasionally large datasets and visualizes it.

REFERENCES

[1] D. Moores, E. Cirino, and William Morrison, MD, "Electromyography (EMG)," March 2018. [Online]. Available: https://www.healthline.com/health/electromyography

[2] J. Brownlee, "Classification Accuracy is Not Enough: More Performance Measures You Can Use," June 2019. [Online]. Available: https://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-measures-you-can-use/

[3] A. Choudhary, "Principal Component Analysis (PCA) with Python," September 2019. [Online]. Available: https://datascienceplus.com/principal-component-analysis-pca-with-python/

[4] J. Brownlee, "Gentle Introduction to Eigenvalues and Eigenvectors for Machine Learning," February 2018. [Online]. Available: https://machinelearningmastery.com/introduction-to-eigendecomposition-eigenvalues-and-eigenvectors/

[5] C. Nicholson, "A Beginner's Guide to Eigenvectors, Eigenvalues, PCA, Covariance and Entropy." [Online]. Available: https://pathmind.com/wiki/eigenvector

[6] B. Ambielli, "Gini Impurity (With Examples)," October 2017. [Online]. Available: https://bambielli.com/til/2017-10-29-gini-impurity/

[7] J. Brownlee, "A Gentle Introduction to the Rectified Linear Unit (ReLU)," August 2019. [Online]. Available: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

[8] R. Roy, "ML | Stochastic Gradient Descent (SGD)." [Online]. Available: https://www.geeksforgeeks.org/ml-stochastic-gradient-descent-sgd/

[9] T. Yiu, "The Curse of Dimensionality," July 2019. [Online]. Available: https://towardsdatascience.com/the-curse-of-dimensionality-50dc6e49aa1e

[10] A. Aghajanyan, D. K. Patchigolla *et al.*, "Does Dimensionality curse effect some models more than others?" December 2015. [Online]. Available: https://stats.stackexchange.com/questions/186184/does-dimensionality-curse-effect-some-models-more-than-others

[11] J. Stoltzfus, "How does Occam's razor apply to machine learning?" [Online]. Available: https://www.techopedia.com/how-does-occams-razor-apply-to-machine-learning/7/33087

Cale Dunlap Master of Computer Science student at Arizona State University's Ira A. Fulton Schools of Engineering.