Get a Grip! An Exploration of Linear Classifiers for Decoding Hand Gestures from Electromyographic Data

Taylor Hansen, Michael Paskett
CS 6350: Machine Learning
12/14/2018

Introduction:

Traditional commercial myoelectric prostheses utilize 1-2 channels of electromyographic (EMG) data to drive the function of prosthetic devices. Different classes of movements can be selected by co-contracting the flexor and extensor mass, and the degree of movement can be controlled by contracting the flexor or extensor mass separately. This form of control is non-intuitive, and may be part of the problem involved in the abandonment of upper-limb prostheses.

In this project, we aim to develop an intuitive method for controlling the movement types of a prosthetic hand. We have used a high channel (32 single-ended + 496 differential) EMG recording system to classify movement classes based on patterns of muscle activity as recorded from the surface of the forearm (Figure 1). Successful implementation of this classifier would provide the prosthetic user with an intuitive method of controlling a prosthetic hand, in which the hand would follow the intent of the user. This projects extends beyond what was covered in the course, requiring multi-class prediction. For this project, we have compared the performance of several multi-class classifiers utilizing different linear algorithms (simple perceptron, support vector machine (SVM), and logistic regression).

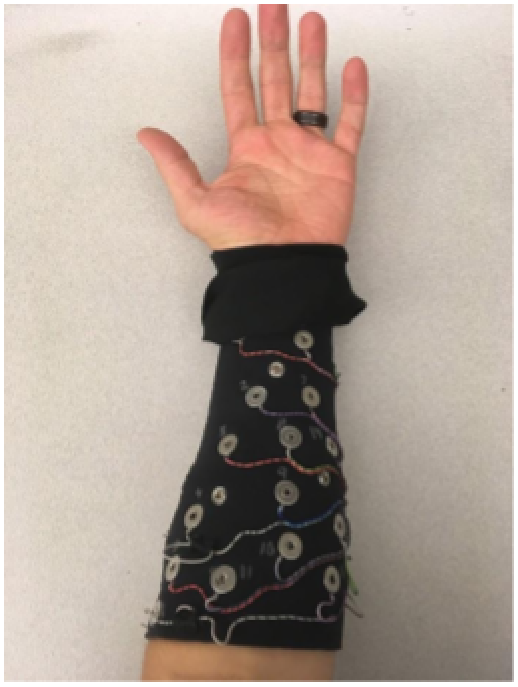
Figure 1 - Surface Electrode Sleeve for Recording Electromyographic Potentials

Methods:

Surface EMG signals were acquired from one intact (non-amputee) subject using the recording system described above. Explicitly, EMG data were collected at 1 kHz via a Grapevine bio-amplification system (Ripple LLC, Salt Lake City, Utah). This system used 15-375 Hz bandpass and 60/120/180 Hz notch filters to filter the data. Following this process, the data were fed into a custom software suite that utilized LabVIEW (National Instruments Corporation, Austin, TX) and multiple instances of MATLAB (Mathworks®, Natick, MA) to process the EMG data and extract features for use with either offline or real-time control algorithms. This particular work focused on offline control decoders, although the intended eventual application is with a real-time classification system.

Data collection was accomplished in a virtual reality environment in which a virtual robotic hand was used to guide the intact subject in performing various hand grasps (Figure 2) (MSMS, MDDF-USC, Los Angeles, California). The hand grasps used were full-flexion grasp, pinch, key grip, tripod pinch, rest, extend all, and point (a total of seven grasps or seven classes). At run time, the virtual robotic hand would move to a randomly assigned grasp with a rise time of 0.7 s, and the subject mimicked the grasp with their physical hand for 3 s. After each trial, the virtual hand would move back to rest, and the subject was instructed to relax their hand, as well. Following a brief inter-trial rest period (1 s), another random grasp would be assigned to the virtual hand, and the subject would change positions to best approximate the new grasp. This process continued until each grasp had been assigned a total of five times. This was equivalent to 35 total grasps in a single data set.

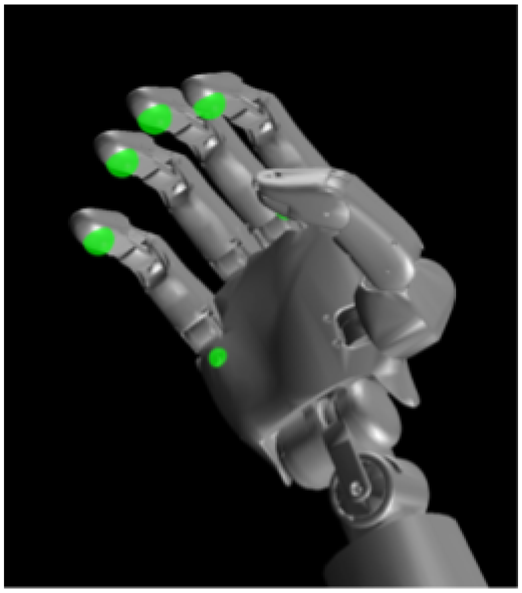
Figure 2 - MSMS Virtual Hand.

Two datasets were collected according to the above regime: a testing set and a training set. The training set was divided for five-fold cross-validation, where one fold was left out as the validation set. Subsequently, each fold was divided further into seven sets enabling the one-vs-all classification scheme (i.e., for a given set, the labels for a single class are positive and the remainder are negative). After the data was prepared for one-vs-all classification, the data was passed into the cross-validation step. The algorithms explored for this project included a simple perceptron, a support vector machine (SVM) with stochastic subgradient descent, and logistic regression with stochastic gradient descent. In the cases of SVM and logistic regression, the initial learning rate was decayed at the t^{th} epoch according to $\gamma_t = \frac{\gamma_0}{1+t}$ where γ is the learning rate.

During the five-fold cross-validation, the algorithms were tested with various hyperparameters (Table 1). If an optimal hyperparameter was found to be on the edge of the set, an additional hyperparameter was tested an order of magnitude beyond the current optimal hyperparameter. Optimal hyperparameters were found for each of the algorithms. Using the optimal hyperparameters, a classifier was trained on the entire training set.

Table 1 - Hyperparameters Tested During Cross-Validation			
	Perceptron	SVM	Logistic Regression
Initial Learning Rate:	$1e^{-1}$, $1e^{-2}$, $1e^{-3}$, $10e^{-4}$, $1e^{-5}$, $1e^{-6}$	$1e^{-1}$, $1e^{-2}$, $1e^{-3}$, $1e^{-4}$, $1e^{-5}$	$1e^{-3}$, $1e^{-4}$, $1e^{-5}$
Epochs:	5:20	5:20	10:20
Regularization/Loss:	N/A	C: $1e^{3}$, $1e^{2}$, $1e^{1}$, $1e^{0}$, $1e^{-1}$, $1e^{-2}$, $1e^{-3}$, $1e^{-4}$	σ^2: $1e^{-2}$, $1e^{-1}$, $1e^{0}$, $1e^{1}$, $1e^{2}$, $1e^{3}$, $1e^{4}$

When predicting labels (i.e. grasps) for the test set, each of the seven one-vs-all classifiers for a given algorithm predicted whether or not the current example should be labeled with its class. For example, the ‘pinch’ classifier would predict whether the current example was a pinch or not, the ‘key grip’ classifier would make a separate prediction, etc. If ambiguities were detected, the prediction defaulted to the previous example’s predicted label. If there was an ambiguity on the first example, the assigned label was ‘rest’. The rationale behind these decisions is that in cases of uncertainty, it is better to maintain the previous grasp. Otherwise, control could be susceptible to jitter. Overall accuracies on the test set were calculated for each algorithm. Confusion matrices were generated for each algorithm, as well, to visualize which classes were most susceptible to misclassification.

Results:

Typical results following five-fold cross-validation are summarized in Table 2. Note that the perceptron, in particular, is susceptible to slight changes in these reported values due to random initialization of the weight vector.

Table 2 - Optimal Hyperparameters and Overall Test Accuracy			
	Perceptron	SVM	Logistic Regression
Initial Learning Rate:	$1e^{-3}$	$1e^{-4}$	$1e^{-3}$
Epochs:	17	12	12
Regularization/Loss:	N/A	C: $1e^{2}$	σ^2: $1e^{1}$
Cross-Validation runtime:	20.4 m	38.7 m	2.45 h
Overall Test Accuracy:	92.16%	94.14%	92.28%

Confusion matrices for perceptron, SVM, and logistic regression are shown in Figures 3, 4, and 5, respectively. SVM was found to have the greatest accuracy on the test set, followed by logistic regression and the perceptron.

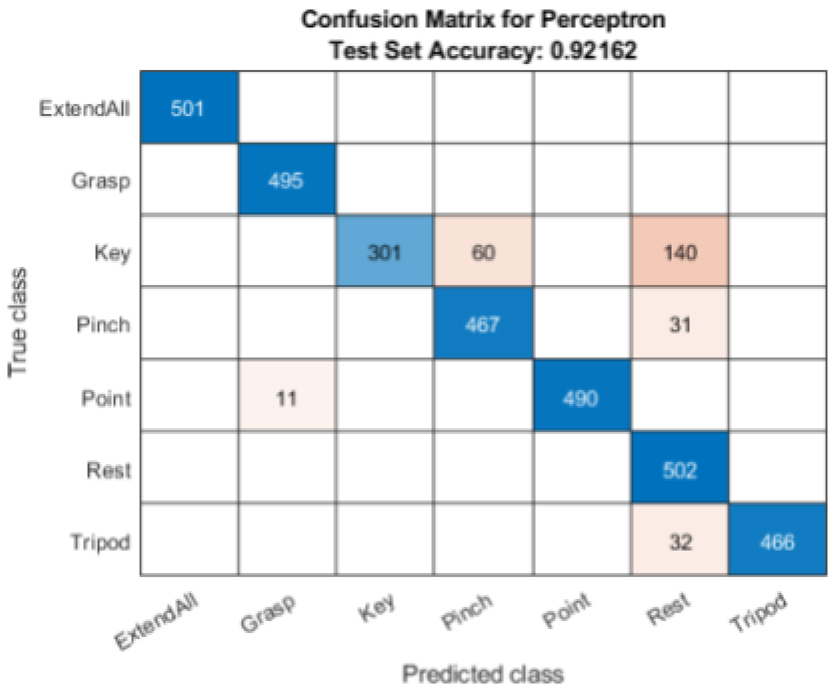

Figure 3 - Confusion matrix for perceptron

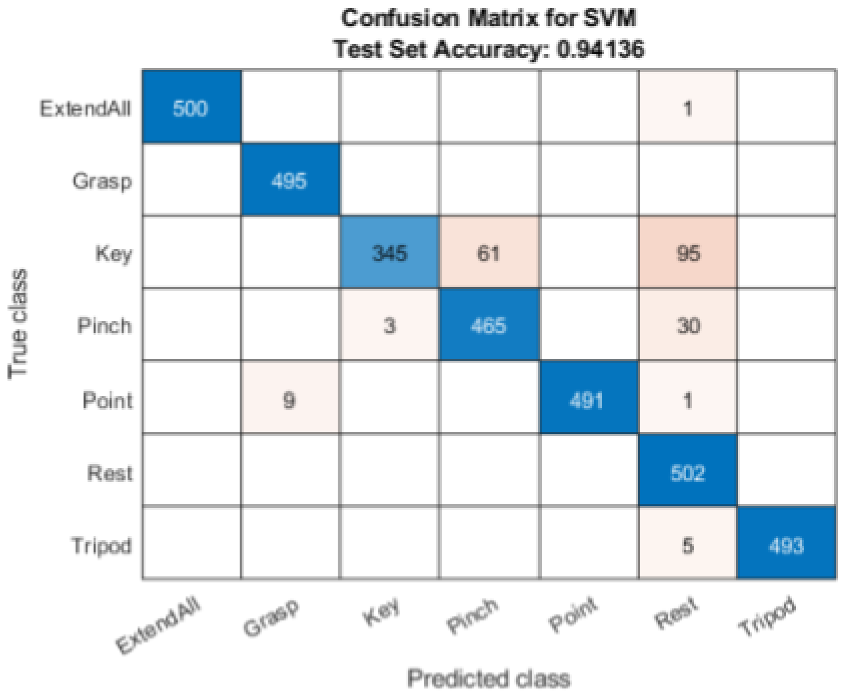

Figure 4: Confusion matrix for SVM

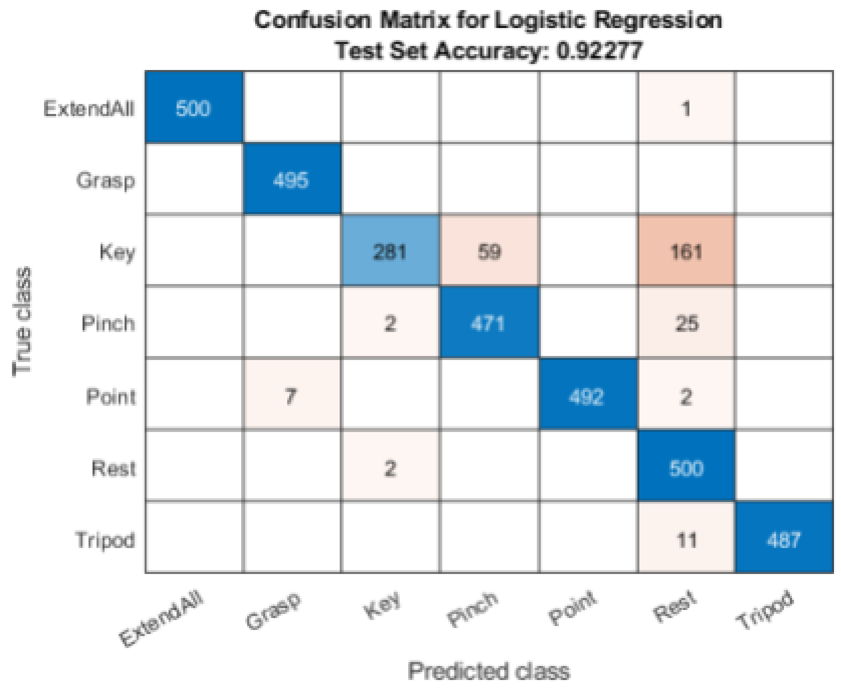

Figure 5: Confusion matrix for logistic regression

Discussion:

Despite their structural differences, the three algorithms performed comparably well on the test set, with the difference between the best performer and the worst performer a mere 2.02% in overall accuracy. Perhaps a more telling metric is the amount of time needed to run cross-validation. While the perceptron and SVM computed for a reasonable 20 m and 40 m, respectively, logistic regression wasted away for nearly 2.5 h with no meaningful boost in performance on the test set. If this large-scale compute during cross-validation is any indication of projected performance in a real-time system, it would be improbable to consider logistic regression as a feasible approach.

Of note is the apparent difficulty in classifying the key grip, which was most often misclassified as rest. As the physical difference between the rest and key grip is small, it seems reasonable for this misclassification to occur. In future work, it would be of interest to discover if nonlinear classifiers might be able to separate the two classes with higher accuracy.

Given much more time on this project, there are multiple things we would continue to explore. Given the similar, albeit high, performance across the three algorithms we explored, we wonder if other algorithms might perform even better. Interesting results might be obtained by exploring neural networks or even an ensemble of the three algorithms we've already implemented. Although we ultimately decided on a one-vs-all multiclass classification approach, we could have taken an all-vs-all approach in which 21 classifiers would be trained per algorithm, instead of seven for our current grasp set. This would provide for more complex prediction using a majority vote or a 'label tournament' of sorts. Ambiguities would likely still be encountered in this approach, too, as were encountered in the one-vs-all approach.

Translating this work to real-time control of a robotic prosthesis will require not only much more algorithmic work but also much more data. In this project, we explored seven common hand grasps, but there are many more classes we could have explored but chose not to for the time being. Our data was also limited to a single intact subject. Validation of the presented results need to be expanded to a statistically significant number of intact individuals. Ideally, this data would be supplemented with amputee data, as well.

Conclusion:

Linear classifiers are capable of predicting classes of hand movements based on high-dimensional EMG data. Using a rudimentary one-vs-all approach for multi-class classification, we were able to classify movements to a high degree of accuracy (>90%) with three linear classifiers. This project will inform future work in incorporating a real-time classifier for the intuitive control of a prosthetic limb.

Special Thanks:

We would like to the Dr. Vivek Srikumar for an enjoyable and thorough introduction to machine learning. The material we learned has been very beneficial for our own research and we are excited to have the background necessary to move forward with this project.