(Tentative title) MYO Armband for gaming applications

Yegor Chsherbatykh
School of Science and Technology
Nazarbayev University
Astana, Kazakhstan
yegor.chsherbatykh@nu.edu.kz

Dinmukhammed Baimurza
School of Science and Technology
Nazarbayev University
Astana, Kazakhstan
dinmukhammed.baimurza@nu.edu.kz

***Abstract*—**

***Index Terms*—MYO, EMG, LDA, Logistic regression, machine learning, gaming**

I. Introduction

Electromyography (EMG) is a bio-medical signal that is produced by the muscular electrical activity in response to stimuli [1]. A muscle is a set of motor units each composed out of muscular cells innervated by a motor neuron. Muscles are connected to tendons which serve to move skeleton structures. A resting potential between the intracellular and extracellular environments of the muscle is about -80mV. Motor neuron initiates muscle contraction by releasing specific neurotransmitters from the corresponding nerve endings. Increased neurotransmitter concentration is picked up by the muscle cell receptors which in turn initiate a process of membrane depolarization. A wave of polarization and depolarization is propagated across the neighboring muscle fibers causing them to contract [1] [2].

The summation of the action potentials from all fibers in a particular motor unit generates a motor unit action potential (MUAP). Thus, the EMG signal can be observed as a result of the summation of multiple MUAPs [3]. Generally, there are two types of sensors used for EMG signal acquisition: invasive and non-invasive. Invasive sensors are inserted directly into the muscle tissue while non-invasive are placed on the skin above the region of interest [1]. For the purpose of this study non-invasive sensor from the MYO Armband will be used.

MYO armband is a human-computer interface developed by Thalmic labs in 2014. The armband is placed at the upper part of the forearm where it acquires EMG activity through 8 sensory modules. The device communicates with the computer by using Bluetooth. MYO armband supports five hand gestures from the box: wave in, wave out, fist and spread fingers. Those gestures can be then mapped to predefined user commands or application. Additionally, real-time raw EMG data can be acquired via the MYO Armband SDK.

In this paper, we will propose a gesture-based framework which would allow real-time usage of the MYO Armband in different gaming scenarios. In the further section, we will provide performance measurements of the framework in relation to the classification model, train-test split ratio and filtering window size.

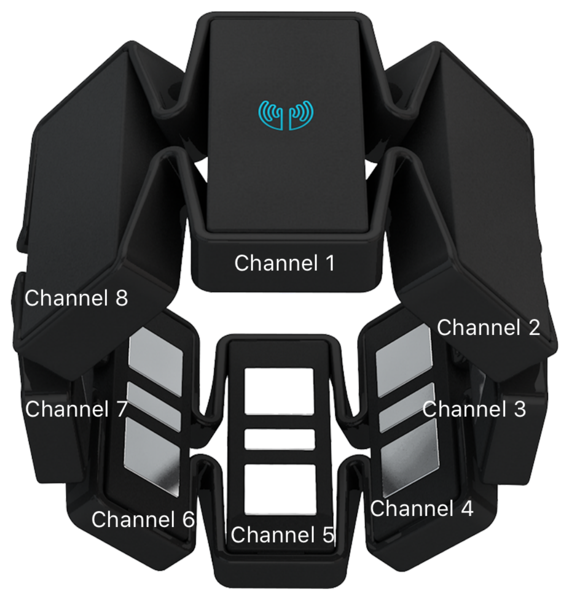

Fig. 1. MYO Armband and labeled channels

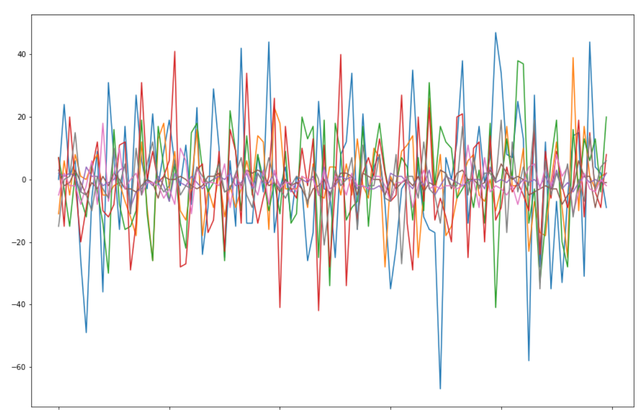

Fig. 2. Raw EMG signal sample

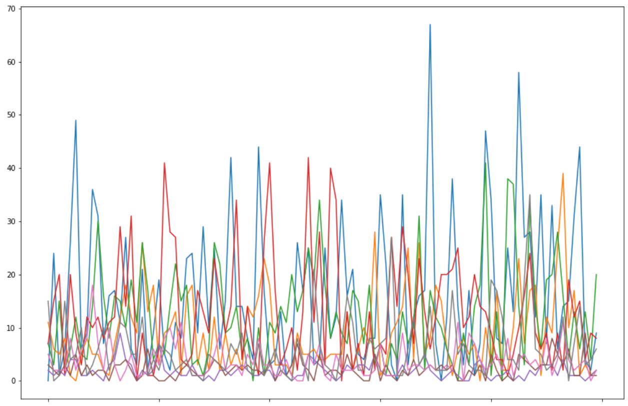

Fig. 3. Rectified EMG signal sample

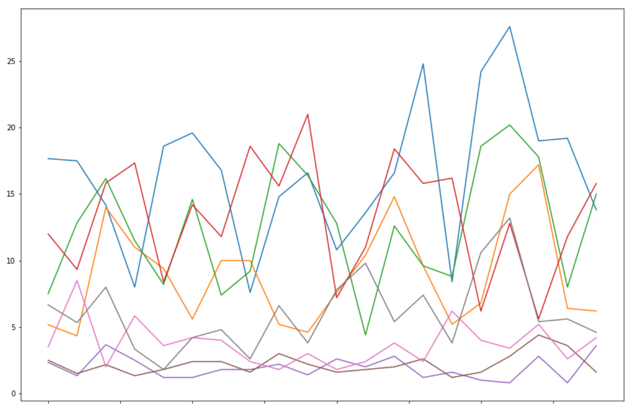

Fig. 4. Filtered EMG signal sample

II. Materials and Methods

A. Materials

In this study we used non-invasive commercial EMG sensor Myo Armband. Small size, portability and relatively low cost, and a proprietary SDK make MYO Armband a great device for quick EMG data acquisition. MYO Armband has 8 dry sensors (Figure 1). Those 8 sensor transfer EMG data through Bluetooth at a rate of 200Hz. The armband should be placed at the upper part of the forearm. Before every experimental trial, MYO had been worn for a period ranging from 5 to 10 minutes for device warm-up and more accurate sensory output. Once the device is connected to the PC, the raw EMG data was acquired by using the official Thalmic Lab SDK.

B. Methods

1) Data acquisition: The raw EMG data from the MYO Armband is collected through the window $W^{N\times 8}$ where $N \subseteq \mathbb{Z}^{+}$ and represents number of EMG samples. The window size is adjustable based on the gaming application used. If the window size is 200 samples ($|W| = 200$). Each column of the W is represented by the channel vector $C_i^{200\times 1}$ received from the channel $i = 1, 2, 3, ...8$.

2) Rectification: Preprocessing is a crucial part of EMG signal classification. Raw EMG signal (Figure 2) should be preprocess to eliminate random high-frequency and low-frequency noise [1]. This is usually done by averaging the signal $S_n \subseteq \mathbb{R}^{N\times 8}$ and performing full-wave rectification. The result of the signal rectification is a signal $abs(S_n)$ (Figure 3) where each data point is an absolute value of itself. Full-wave rectification preserves the original EMG dynamic and helps to avoid the situation when the EMG signal can be averaged to zero.

3) Filtering: For real-time EMG filtering the moving average algorithm is used. Each channel $C_n \subseteq \mathbb{R}^{N\times 1}$ of the signal S_n is filtered by unweighted mean of the previous n data points p_i.

$$\overline{C_n} = \frac{p_M + p_{M-1} + p_{M-2} + ... + p_{M-(n-1)}}{n}$$

$$\overline{C_n} = \frac{1}{n}\sum_{i=0}^{n-1} p_{M-i}$$

For offline performance estimation of the classifier models the moving average approach is inadequate since it makes data points dependent of each other. Therefore, to avoid the possible dependency between the train and test sets each channel C_n of the signal S_n is averaged by the number of points $N_p \subseteq \mathbb{Z}^{+}$ (Figure 4).

4) PCA: Principle Component Analysis (PCA) is a very powerful yet simple dimensionality reduction technique. The main idea behind PCA is to project data from a high-dimension space onto low-dimension space with maximum variance. The approach is particularly useful for preliminary analysis of the complex multichannel EMG datasets. The result of the PCA is a set of orthogonal principal components among each the variance of the features is maximized and noise is reduced [4].

The PCA uses the notion of variance and covariance extensively. The variance is measure of the data is spread from the mean in a dataset:

$$\text{Var(X)} = \frac{\sum_{i=0}^{n} (X_i - \overline{X})^2}{n-1}$$

Covariance is a measure of how the data dimension vary from the mean with respect to each other. For matrices X and Y the covariance is calculated as follows:

$$\text{Cov(X, Y)} = \frac{\sum_{i=0}^{n} (X_i - \overline{X})(Y_i - \overline{Y})}{n-1}$$

Although the covariance must be calculated only between two dimensions, it is still possible to get the covariance estimates from a multidimensional dataset. Thus, for the feature matrix $X^{N\times 8} = (X_1, X_2, ..., X_8) \subseteq \mathbb{R}^{N\times 8}$ all possible inter-dimensional covariances are calculated and put into covariance matrix $C^{8\times 8}$.

$$C^{8\times 8} = (c_{i,j} = Cov(Dim_i, Dim_j))$$

Before calculating the covariance matrix for the dataset, it is important to subtract mean value $\overline{x}$ from x to obtain a zero mean in all dimensions.

Next step is find eigenvalues and eigenvectors of the covariance matrix Cov. The biggest eigenvalue with a corresponding

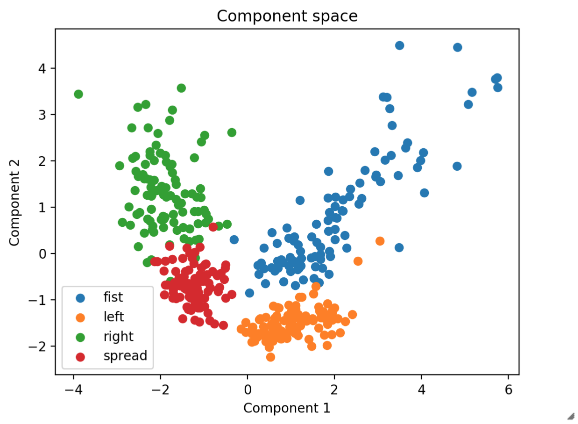

Fig. 5. Four gesture dataset projected on two principle components

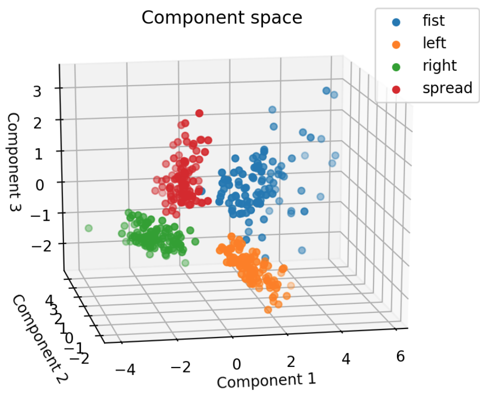

Fig. 6. Four gesture dataset projected on three principle components

eigenvector will be a principle component of the dataset. It is up to us to decide how many principle components to choose from a ComponentVector.

$$\text{ComponentVector} = [eig_1, eig_2, eig_3, ...]$$

The final step is to project the dataset along the chosen principal components. This is done via simple matrix multiplication where both the principle components matrix $P_c \subseteq \mathbb{R}^{8\times 8}$ and feature matrix $X \subseteq \mathbb{R}^{N\times 8}$ are both transposed.

$$\text{ProjectedData} = P_c^T X^T$$

The PCA results on the dataset with 4 prerecorded gestures (wave left, wave right, clenched fist, spread hand) are shown on Figure 5 and Figure 6.

5) Classification: As it has been said before, MYO Armband has 5 default pre-calibrated gestures that it is able to recognize. However, in our research, we trained classification models on custom gestures from raw data in order to accomplish two tasks. First is to estimate the accuracy of MYO armband and our model and second is to have a specifically chosen gestures for a particular game. This approach made a gaming experience better.

We compared two classification algorithms that fits best to our needs. They are Linear Discriminant Analysis and Logistic Regression classification algorithms. Linear Discriminant Analysis or LDA is a machine learning algorithm that is used for a dimensionality reduction or classification. This algorithm is a generalization of Ronald Fisher's linear discriminant method. LDA is close to the PCA in a sense that it tries to show linear combination of variables that explains the given data best, but it focuses on maximizing the separability between the classes. LDA makes it by creating a new axes that are maximizing the mean class difference and minimizing the variance in each class. Such approach gives a good separation between the classes and makes this algorithm to be used as a classifier. Classes in our experiments are individual gestures.

For multiclass classification, LDA measures mean not between the center of two classes, but between the center of N classes and center of each class.

For both classification algorithms we used 1000×8 data points for estimating their performance. We had three experiments on estimating the optimal window size, minimum number of training data and window size vs training data for both algorithms. On average, Logistic Regression has shown better results in all three cases, therefore we chose it to use in our gaming framework.

[7].

C. *Gaming framework*

MYO Armband mobility and ease of use make it a great device for designing new human-computer interfaces and applications. Real-time gesture classifier models produce discrete numerical output that can be used in different computer interactions. For the purpose of this study, a MYO mediated gaming framework was created.

The process of recording EMG gesture data and creating a classification model was automated with a console-based Python application [5]. The model was used in real-time EMG processing application written also in Python [6]. Application was supplied with a classification model in .sav format and a JSON configuration with gesture keyboard mappings. Raw EMG data in a form of vectors $V \subseteq \mathbb{R}^{1\times 8}$ was received at a rate of 200Hz and stored in a queue data structure of size $N \subseteq \mathbb{Z}^{+}$, forming a feature matrix $X \subseteq \mathbb{R}^{N\times 8}$. X was then fed into the classifier which produced classes vector $Y \subseteq \mathbb{Z}^{1\times N}$ where Y_i corresponded to a predicted class for a feature vector V_i. Mode of the vector Y is then mapped to the keyboard input using PyAutoGUI library. Sample mapping configurations are presented below and examples of gaming applications are presented on Figure 11.

Class label	Gesture name	Key
1	Wave-in	left
2	Wave-out	right
3	Clenched fist	enter
4	Spread fingers	space

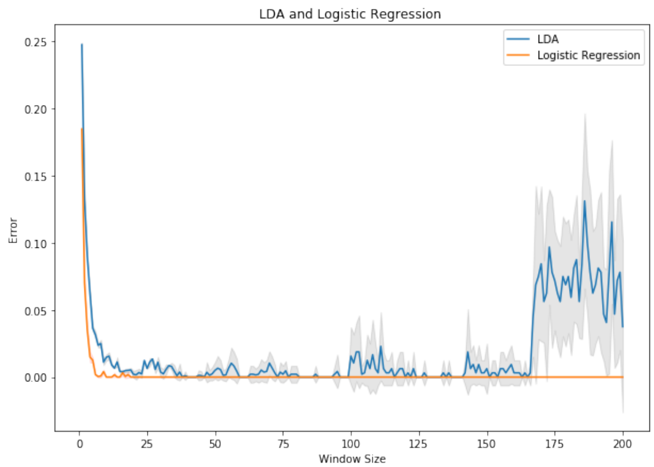

Fig. 7. LDA's and Logistic regression's error in relation to the filtering window size

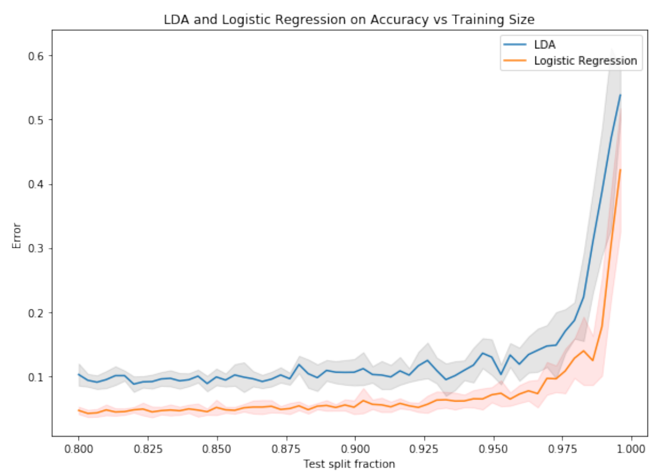

Fig. 8. LDA's and Logistic regression's error in relation to the training size

III. Results

After all classification model evaluations, Logistic regression algorithm appeared to be more effective than LDA both on varying window size and quantity of training data.

In addition to that, we estimated what would be the best combination of parameters of window size vs training data and on this time Logistic Regression again has shown better results. From Fig. 9 and 10 we can see that optimal solution for both algorithms would be a usage of window size not less than 10 points and as much training data as possible.

IV. Discussion and conclusion

Our experiment has shown that optimal algorithm for discreet gesture recognition for MYO Armband is Logistic Regression. For the future work, we can expand our experiment in order to estimate the maximum possible gestures that MYO is able to recognize as well as understand what are the "optimal" gestures that give highest accuracy values on testing. Moreover, we can estimate what is the highest frequency on

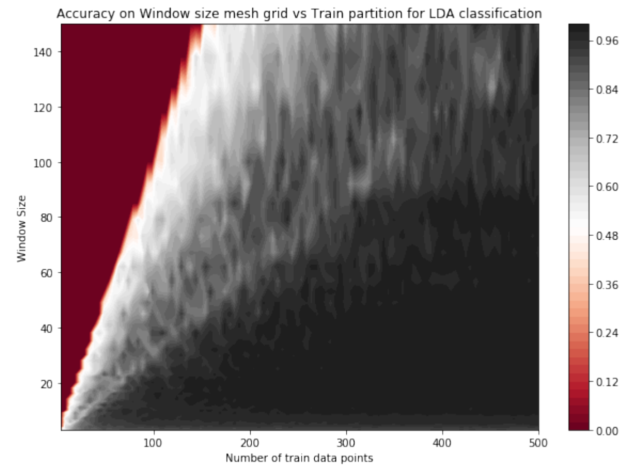

Fig. 9. Window size vs Training data points for LDA classification

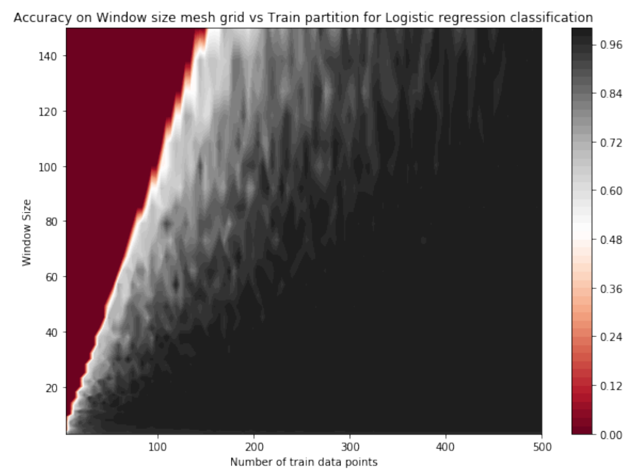

Fig. 10. Window size vs Training data points for Logistic Regression classification

Fig. 11. Examples of gaming scenarios: HexGL on the left and Arkanoid on the right.

which MYO is able to recognize gestures. In addition to that, we can also modify our experiment in order to estimate the performance of MYO on recognition of continuous data, it's response time and it's accuracy as well.

REFERENCES

[1] M. B. I. Reaz, M. S. Hussain, and F. Mohd-Yasin, "Techniques of EMG signal analysis: detection, processing, classification and applications (Correction)," Biological Procedures Online, vol. 8, no. 1, pp. 163163, 2006.

[2] Kuo IY, Ehrlich BE, "Signaling in muscle contraction," Cold Spring Harb Perspect Biol. 2015;7(2):a006023. Published . doi:10.1101/cshperspect.a006023

[3] D. F. Stegeman, J. H. Blok, H. J. Hermens, and K. Roeleveld, "Surface EMG models: properties and applications," Journal of Electromyography and Kinesiology, vol. 10, no. 5, pp. 313326, 2000.

[4] G. Bosco, "Principal Component Analysis of Electromyographic Signals: An Overview", The Open Rehabilitation Journal, vol. 3, no. 1, pp. 127-131, 2010.

[5] Yegor Chsherbatykh, Dinmukhammed Baimurza "yegorsch/myo-validation v1". Zenodo, 24-Nov-2018. [Accessed: 24- Nov- 2018].

[6] Yegor Chsherbatykh, Dinmukhammed Baimurza "yegorsch/EMG-Project v1". Zenodo, 24-Nov-2018.

[7] Fisher, R. A. (1936). "The Use of Multiple Measurements in Taxonomic Problems". Annals of Eugenics. 7 (2): 179188. doi:10.1111/j.1469-1809.1936.tb02137.x. hdl:2440/15227.