Hand Gesture Classification using sEMG Data: Combining Gesture Detection and Cross-Validation

Gabriel S. Chaves
PEE/COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
gabriel.chaves@smt.ufrj.br

Anderson S. Vieira
PEE/COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
anderson.vieira@smt.ufrj.br

Markus V. S. Lima
Poli & PEE/COPPE
Universidade Federal do Rio de Janeiro
Rio de Janeiro, RJ, Brazil
markus.lima@dee.ufrj.br

***Abstract*—Prosthetic hands play a vital role in the rehabilitation of upper limb amputees. Gesture recognition using surface electromyography (sEMG) data has emerged as an excellent option for controlling such prosthetic devices since one does not require invasive methods to obtain these data. In order to improve gesture recognition, we must extract the muscle activity from the raw data before classification, as each gesture has its own patterns. In this paper, we use an artificial neural network classifier with individualized data segmentation based on gesture detection to identify six hand movements. We used data from ten healthy volunteers. By combining data segmentation and cross-validation, we were able to refine the amplitude thresholds used to determine the beginning and end of muscle contractions for each person. We designed several experiments using different types of cross-validation. The performance achieved by the proposed model using 4-fold cross-validation was $(93.6 \pm 0.7)\%$, which represents 3.5% more than the mean accuracy of the baseline model, in which there is a single arbitrarily-chosen segmentation threshold for all volunteers.**

***Index Terms*—Hand gesture classifier, classification, segmentation, cross-validation, sEMG data, artificial neural network**

I. Introduction

The hand gesture recognition problem consists of detecting and classifying the muscle contraction from a hand movement. The solution for such a problem can improve human-machine interface (HMI) applications, such as robotic hand control [1], game interface [2], augmented reality [3], and active prostheses [4]. The last one is of great concern since the loss of a limb can be traumatic, impacting a person's body, emotions, relationships, vocation, and way of life [5]. In the case of transradial amputation, a prosthetic hand can help a person to better adapt to the new life. Hence, the usage of myoelectric prostheses has gained relevance recently [6], [7].

Myoelectric devices are controlled by electromyography (EMG) signals generated by muscle contractions. For such a prosthesis to be suitable, the residual limb must possess measurable EMG signals [6], [8]. In most cases, the range of movements is limited to open and close hands. The user can increase the complexity of the gestures through training in sequential control strategies. However, controlling this prosthesis can be an arduous task, requiring a high level of skill and long training sessions [8]. These factors contribute to a higher rejection rate of upper extremity prostheses than lower extremity ones [9]. To ease the rejection, data acquired through non-invasive approaches, such as surface electromyography (sEMG) signals, are highly recommended for HMI applications [10].

The sEMG is the spatial and temporal interference pattern of the electrical activity of the motor neuron and several muscle fibers [11]. This electrical pulse is the response of the motor neuron when it receives a movement command from the brain [11]–[14]. In other words, the movement of a hand is made up of various muscle stimuli, and an sEMG sensor can capture them over the person's skin. Therefore, it can be used to drive an external device, such as in the prosthetic domain. Another advantage of these sensors is their easiness of operation, which makes possible the existence of many devices that are simple and with high reliability [15]–[17]. However, a common problem in the recognition of such an sEMG is identifying the moment in which the muscle activity begins and ends. Due to the stochastic and non-stationary nature of the sEMG, extracting such a time range is quite challenging. To address this, we segment the sEMG to separate the gesture of interest from the data signal. We can categorize the sEMG segmentation techniques as sliding windows or gesture detection. By using sliding windows, the sEMG is segmented into several time windows that can be adjacent or overlapped. In gesture detection, we extract only muscle contraction from the sEMG data, identifying the beginning and end of muscle activity.

The sEMG-based hand gesture recognition can be divided into five main modules [18]. The data acquisition is the first module, responsible for capturing and reading the data examples. In this module, we decide how many gestures and sensors will be used during the experiments. The preprocessing module segments the sEMG, that is, extracts the hand movement data from the sEMG signal. The feature extraction/selection module selects a proper set of features relevant to the gesture recognition problem. The classification module is where the classifier operates by matching the features to their respective classes. Recently, new machine learning approaches have achieved excellent performance in the design of natural gesture interfaces [10], [18]–[20], whose methods aim to identify muscle contraction patterns in the sEMG signal and match them with predefined gestures through supervised learning [19]. At last, the post-processing module refines the classification output, which helps one to interpret and evaluate the model results.

In [21], the authors achieved 94.0% accuracy in identifying four hand postures using k-nearest neighbor (kNN) and an adjacent sliding window approach to segment the data. The solution proposed in [22] used Gaussian mixture models (GMM), hidden Markov models (HMM), and a combination of overlapped sliding windows with gesture detection in order to classify six hand movements, achieving 99.0% accuracy. The authors of [23] recognized nine hand gestures that were classified with the linear discriminant analysis (LDA) method and an overlapped sliding window approach, attaining an accuracy of 92.2%. In [24], by using a support vector machine (SVM) and gesture detection to classify seven hand movements, the authors reached an accuracy of 89.2%. We also have reports of modern machine learning techniques that have achieved great results in the gesture classification problem. The authors in [25] reached 93.0% accuracy using a convolutional neural network (CNN) and adjacent sliding windows to classify eight hand gestures. Another example is the solution in [26], where the authors achieved an accuracy of 97.2% using CNNs and overlapped sliding windows to recognize seven gestures.

In this paper, we use [18] as the baseline model, because it is a recent work coming from an active research group and whose datasets are publicly available. We propose an ANN to classify six hand movements from 10 people. We study the segmentation process and its relevance to the gesture recognition problem. We also apply the k-fold cross-validation methodology to fine-tune the hyperparameter necessary in such a processing [27], [28].

This work is organized as follows. Section II describes the datasets and classification system used in this study. In Section III, we describe the pre-processing, as well as the segmentation process. Section IV presents the results of the models' performance. Finally, the conclusion is drawn in Section V.

II. Datasets and Classification Model

In this section, we describe the database. We also briefly introduce the classification model used to identify hand gestures. We can define the present classifier by dividing it into five modules: data acquisition, pre-processing, feature extraction/selection, classification, and post-processing. Each module is responsible for particular tasks, which we will introduce in this section.

A. Datasets

We use the public datasets created by [18]. The data signals were recorded using the Myo armband, a commercial low-cost EMG sensor built by Thalmic Labs. By wearing this device on the forearm, the sensor returns the digital EMG signal from the flexor, pronator, and extensor muscles to a computer via Bluetooth [20]. The resulting Myo armband EMG signal is a combination of 8 channels operating at a sampling rate of 200 Hz. Each channel outputs the measurement of a sensor that makes up the armband. It is worth mentioning that the sensors are not placed on specific muscles. Instead, they are distributed to cover the surface of the forearm.

The datasets are comprised of EMG signals from 6 hand gestures. Among those, the gestures of interest are fist (*fi*), wave-in (*wi*), wave-out (*wo*), open (*op*), and pinch (*pi*). The sixth gesture is the rest position, representing the absence of any muscle activity. Although its classification is not of interest to us, we use many of its characteristics to help distinguish the various EMG patterns of interest. We use the label no-gesture (*no*) for all the gestures our model cannot classify or the non-interesting ones, including the rest position.

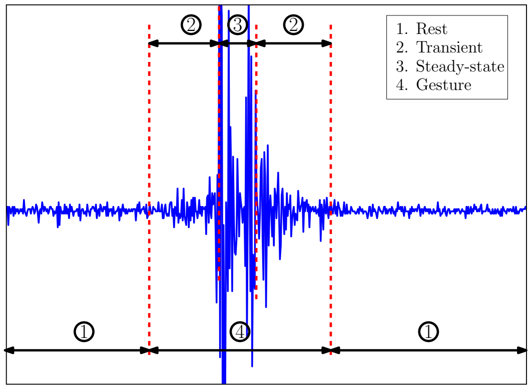


Fig. 1: The gesture structure of a one-channel sEMG example of the database.

The database comprises data signals acquired from 10 healthy volunteers. For each data example, volunteers were instructed to perform specific hand movements. Each example consists of an eight-channel sEMG signal representing continuous hand movements: rest, the final position of interest, and rest again. Each example also carries a single label corresponding to the final position of interest. Fig. 1 illustrates a typical sEMG present in the database, demonstrating amplitude variations based on the types of movement of the hand. The amplitude remains low during resting and increases when the user is executing the gesture. Our classification objective is to classify the signal segment marked as (4) in the figure, representing a gesture. This gesture is considered a random process comprising transient and steady-state components [11], [12], [18], [29], [30].

The *training dataset* $\mathcal{D}_{\text{train}} = \{(\underline{\mathbf{F}}_1, Y_1),\ \ldots\ ,(\underline{\mathbf{F}}_N, Y_N)\}$ of each person contains a total of $N = 30$ data examples, where the matrix $\underline{\mathbf{F}}_i \in [-1,1]^{400\times 8}$ corresponds to the ith EMG signal measured. Each $\underline{\mathbf{F}}_i$ has a duration of approximately 2 s. Fig. 2 shows one column of the signal $\underline{\mathbf{F}}_i$. The categorical variable $Y_i \in \{1,2,3,4,5,6\}$ denotes the label for the signal $\underline{\mathbf{F}}_i$, with $i = 1, 2,\ \ldots\ , N$. Table I shows the relation between Y_i and the gestures of interest. Each one of the classes has 5 examples, resulting in a total of 30 training examples for each user's dataset. Given that the database comprises data

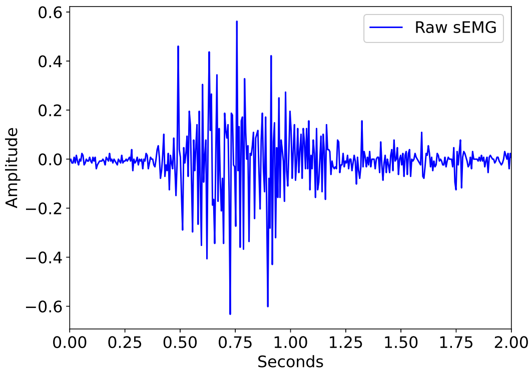


Fig. 2: The sEMG representing one channel of an example $\underline{\mathbf{F}}_i$ from the training dataset $\mathcal{D}_{\text{train}}$.

from 10 subjects, we have 300 training data examples in total. It is important to note that the users maintained their hands in the rest position throughout the measurement period when recording the *no-gesture* class.

TABLE I: Indices and the corresponding hand gestures

Index Y	**Gesture/Class**
1	"rest/no-gesture" (*no*)
2	"fist" (*fi*)
3	"wave-in" (*wi*)
4	"wave-out" (*wo*)
5	"open" (*op*)
6	"pinch" (*pi*)

The *testing dataset* $\mathcal{D}_{\text{test}} = \{(\underline{\mathbf{G}}_1, \hat{Y}_1), \ \ldots \ , (\underline{\mathbf{G}}_M, \hat{Y}_M)\}$ of each user is made by $M = 150$ examples, each recorded for a duration of 5 seconds. An important characteristic of the testing sets is the absence of the no-gesture class. Due to that, the testing set $\mathcal{D}_{\text{test}}$ has 30 examples of each class of $\hat{Y}_i \in \{2, 3, 4, 5, 6\}$, culminating in 150 testing examples for each user. Each matrix $\underline{\mathbf{G}}_i$ within the testing set is an element of the space $[-1, 1]^{1000\times 8}$, with $i = 1, 2, \ \ldots \ , M$. The test sEMGs are longer than the training ones because it is easier to position the EMG segment that corresponds to the gesture in any region within the 5 s of measurement. Fig. 3 depicts one column of the signal $\underline{\mathbf{G}}_i$, the data from $\mathcal{D}_{\text{test}}$.

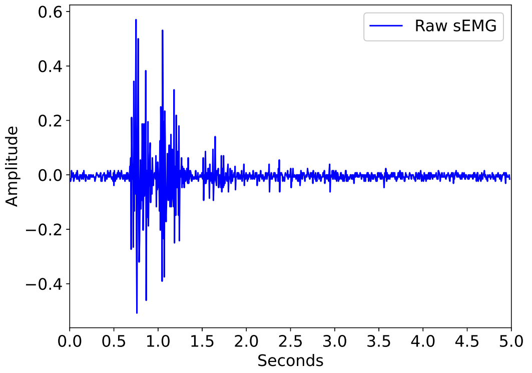


Fig. 3: The sEMG representing one channel of an example $\underline{\mathbf{G}}_i$ from the testing dataset $\mathcal{D}_{\text{test}}$.

B. Classification Model

The *data acquisition* is the first module of the system. It is responsible for reading the data signals from the datasets. In this module, we define a time window $W = (w_1, \ \ldots \ , w_m)$, where $w_i \in \mathbb{Z}^+$, with $i = 1, 2, \ \ldots \ , m$ [11], [31]–[35]. In the training stage, we set the length m_{train} of the window W as $m_{\text{train}} = 400$ points, that is, a time window that selects all points of each training example. In the testing stage, we choose $m_{\text{test}} = 500$, representing half of the total data points in each testing example. Thus, the current module outputs observations represented by the matrix $\underline{\mathbf{E}} = [\mathbf{E}_1; \ \ldots \ ; \mathbf{E}_8] \in [-1, 1]^{m\times 8}$, where the column vector $\mathbf{E}_i = [E_{i1}, \ \ldots \ , E_{im}]^{\mathrm{T}}$ corresponds to the ith channel of the observation $\underline{\mathbf{E}}$, with $i = 1, 2, \ \ldots \ , 8$.

The *pre-processing* is the second module in the classifier. In this module, we perform some mathematical operations in each observation window $\underline{\mathbf{E}}$ outputted by the previous module. The main goal of the pre-processing is to segment the sEMG signal. Such a process extracts only the portion of data related to muscle contraction. The output of the pre-processing module is $\underline{\mathbf{Z}}$.

The *feature selection* is the third module of the classification model, and is crucial for machine learning approaches. Since the sEMG is a spatial and temporal interference pattern of the muscular electrical activity near the detection surface, we need a feature capable of synthesizing a large amount of temporal and spatial data. Dynamic time warping (DTW) is a technique that can "align" two time series by returning the minimal cost to match such signals, i.e., the DTW outputs the "distance" between two time series [18], [36]. In other words, it can store the similarity between two functions independently of their distances in time or space. In this module, we first calculate the DTW between all the pre-processed training examples with the same labels. This approach generates the DTW distances between the training examples inside the same class. Next, we choose the pre-processed signal $\underline{\mathbf{H}}_i^* \ \forall \ i = 1, \ \ldots \ , 6$ that is closest to all elements in each class. Finally, the module outputs the feature vector for the signal $\underline{\mathbf{Z}}$ as $\mathbf{X} = (\mathrm{dtw}(\underline{\mathbf{H}}_1^*, \underline{\mathbf{Z}}), \ \ldots \ , \mathrm{dtw}(\underline{\mathbf{H}}_6^*, \underline{\mathbf{Z}}))$, where dtw denotes the DTW distance for multichannel time series.

The *classification* module is where we predict the gestures. This module receives the feature vector $\mathbf{X}$ as an input and outputs the predicted gesture Y. The predictive model is given by the following equation

$$\psi(\mathbf{X}) = \operatorname*{argmax}_{y\in\{1,2,3,4,5,6\}} \mathbb{P}(Y = y|\mathbf{X}) \ , \tag{1}$$

subject to the constraint that the conditional probability that maximizes (1) is equal to or greater than λ, i.e., the minimum probability for $\mathbf{X}$ be classified. Otherwise, $\mathbf{X}$ always receives the no-gesture label. In this work, we used a feedforward neural network (FNN). We opt for a shallow FNN having only three layers: input, hidden, and output. We used 6 neurons in each layer, so the neural network topology is $6\times6\times6$. It should be noted that the input layer receives the signal $\mathbf{X}$ standardized by $(X_j-\mu)/\sigma$, where μ and σ denote the mean and standard deviation of $\mathbf{X}$, respectively, with $j=1,2,\ \dots\ ,6$.

Post-processing is the last module of the system model. It is responsible for processing the data before presenting it to the user. This module processes the classifier's response by eliminating consecutive labels. We use a time delay of one observation for this task. For example, the classification module outputs two labels $\psi(\mathbf{X})_{k-1}, \psi(\mathbf{X})_k$ for some $k\in\mathbb{Z}^+$. When the label $\psi(\mathbf{X})_k$ is equal to $\psi(\mathbf{X})_{k-1}$, the current module returns $\psi(\mathbf{X})_k = no$. Otherwise, we return the label $\psi(\mathbf{X})_k$ unchanged.

III. Segmentation Processing

In this section, we propose some modifications to the original study. As stated in the previous section, the authors in [18] do not explore some parameter adjustments. For example, in [18], the threshold responsible for the muscle activity detection τ_u has the same value for all subjects.

A. Gesture Detection

There are several approaches to segmenting an sEMG signal, such as sliding windowing and gesture detection. Sliding windowing methods use windows of fixed length to segment the data at each moment, in which they can overlap or be adjacent to each other. However, we used a gesture detection technique. Such a method identifies the beginning and end of the muscle activity, resulting in the signal range where there is only the desired information.

We start by defining the input signal as the matrix $\underline{\mathbf{E}} = [\mathbf{E}_1;\ \dots\ ;\mathbf{E}_8] \in [-1,1]^{m\times8}$, where the column vector $\mathbf{E}_i = [E_{i1},\ \dots\ ,E_{im}]^{\mathrm{T}}$ corresponds to the ith channel of $\underline{\mathbf{E}}$, with $i=1,2,\ \dots\ ,8$, and m is the number of points acquired by a time window W. In the first step, we take the absolute values of $\underline{\mathbf{E}}$ by rectifying it, which results in the new signal $\underline{\mathbf{R}} = \text{abs}(\underline{\mathbf{E}}) \in [0,1]^{m\times8}$. The second step consists of filtering $\underline{\mathbf{R}}$ with a low-pass Butterworth filter Φ with a cutoff frequency of 10 Hz [37]–[40]. This process yields the signal $\underline{\mathbf{V}} = \Phi(\underline{\mathbf{R}}) \in [0,1]^{m\times8}$. Fig. 4 shows the envelope $\mathbf{V_i}$ of the one-channel signal of $\underline{\mathbf{F}}_i$ in Fig. 2. We decrease the dimensions of the data in the third step by summing along the channel axis of $\underline{\mathbf{V}}$, which results in $\mathbf{S} = \text{sum}(\underline{\mathbf{V}}) \in [0,8]^{m\times1}$. In the fourth step, we analyze the signal in the frequency domain. We use the short-time Fourier transform (STFT) to compute the spectrogram $\underline{\mathbf{P}}_C = \mathcal{S}(\mathbf{S}) \in \mathbb{C}^{26\times p}$ of $\mathbf{S}$, dividing the sampling frequency interval $[0,100]$ Hz into 26 points, where $p = \text{floor}((m-10)/15)$ and m is the length of $\underline{\mathbf{E}}$, $\underline{\mathbf{R}}$ and $\underline{\mathbf{V}}$ [41]. We take the modulus of $\underline{\mathbf{P}}_C$ and sum its columns, obtaining the vector $\mathbf{U}\in\mathbb{R}^{26\times1}$ in the fifth step.

The last step consists in extracting the gesture of interest muscle activity sEMG. We denote by i_s as the beginning of the hand movement and i_e as the end. It is worth highlighting that both indices are obtained from $\mathbf{U}$. We analyze the entries of $\mathbf{U}$ and save the indices where the difference of two consecutive values is equal or greater than τ_u. Thus, we set i_s and i_e as the first and second indices, respectively. We also determine a fixed range of points a where $i_e - i_s$ must be greater. In other words, if the difference $i_e - i_s$ is less than a points, then the segmentation returns the signal $\underline{\mathbf{Z}} = \underline{\mathbf{V}}$. Otherwise, we segment $\underline{\mathbf{V}}$, obtaining $\underline{\mathbf{Z}} = \underline{\mathbf{V}}(i_s : i_e, :)$. One can notice that

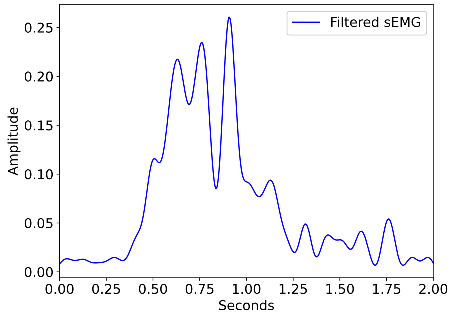


Fig. 4: Filtered signal $\mathbf{V_i}$ of an one-channel rectified sEMG $\mathbf{R_i}$.

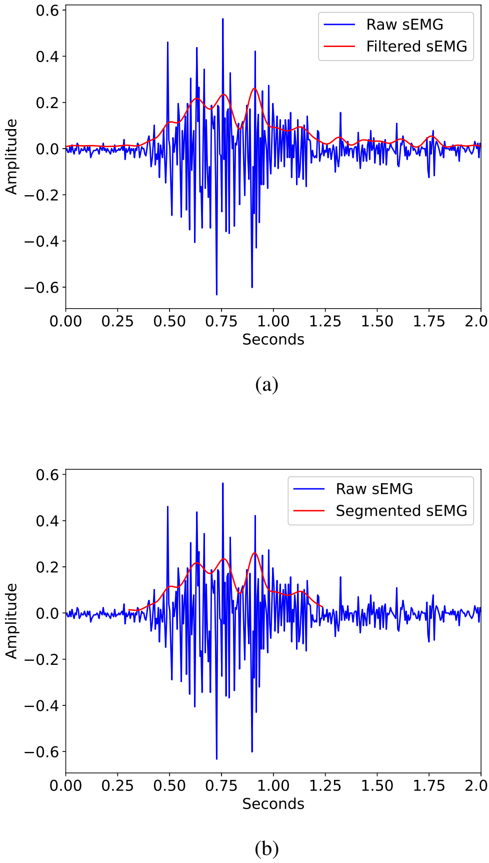


Fig. 5: One-channel sEMG data signal and its respective preprocessed version: (a) Filtered; (b) Segmented.

such a segmentation procedure outputs signals with different lengths, depending on the duration of muscle activity. Next, if $i_s = 1$ and $i_e = m$, we return the no-gesture label and proceed directly to the post-processing module. Otherwise, we send the resulting signal $\underline{\mathbf{Z}}$ to the next module. One can notice that the pre-processing module will output an unsegmented signal in two cases: the muscle activity is shorter than a points, and when the module cannot segment the muscle contraction at all. Fig. 5 depicts the one-channel sEMG signal compared to its filtered and segmented versions. Fig. 6 summarizes all the operations until segmentation,

```
SEGMENTATION
Rectify $\underline{\mathbf{E}} \rightarrow \underline{\mathbf{R}} = \text{abs}(\underline{\mathbf{E}}) \in [0,1]^{m\times 8}$
Envelope $\underline{\mathbf{E}} \rightarrow \underline{\mathbf{V}} = \Phi(\underline{\mathbf{R}}) \in [0,1]^{m\times 8}$
$\Phi$ is a low-pass Butterworth filter
Segmentation of $\underline{\mathbf{V}}$:
  • Sum along the channel axis $\rightarrow$ $\mathbf{S} = \text{sum}(\underline{\mathbf{V}}) \in [0,8]^{m\times 1}$
  • Spectrogram $\rightarrow$ $\underline{\mathbf{P}}_C = \mathcal{S}(\mathbf{S}) \in \mathbb{C}^{26\times p}$
  • Modulus of $\underline{\mathbf{P}}_C$ $\rightarrow$ $\underline{\mathbf{P}} \in \mathbb{R}^{26\times p}$
  • Sum along the columns of $\underline{\mathbf{P}}$ $\rightarrow$ $\mathbf{U} = \text{sum}(\underline{\mathbf{P}}) \in \mathbb{R}^{26\times 1}$
  • Detect the muscle activity by finding $i_s$ and $i_e$ where $\mathbf{U} > \tau_u$
  • If $(i_e - i_s) < a$:
      ▫ $\underline{\mathbf{Z}} = \underline{\mathbf{V}}$
      ▫ Else if $i_s = 1$ and $i_e = m$:
          × $\underline{\mathbf{V}}$ receives the no-gesture classification
          × Else:
              ◦ Segment $\underline{\mathbf{V}}$ $\rightarrow$ $\underline{\mathbf{Z}} = \underline{\mathbf{V}}(i_s : i_e, :)$
```

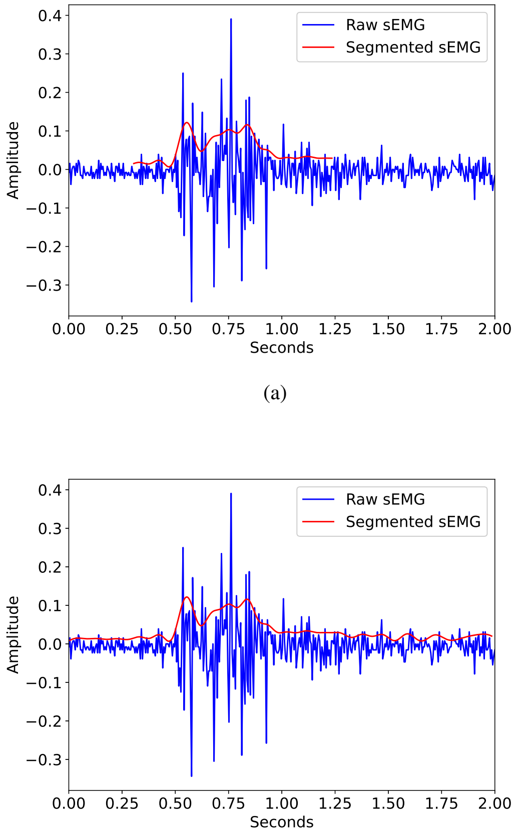
Fig. 6: Block diagram of the segmentation processing.

B. *Muscle Activity Detection Threshold*

Choosing the muscle activity threshold value τ_u can be a challenging task. Such a value depends on the entries of the array $\mathbf{U}$, which is composed of low- and high-magnitude components that refer, ideally, to the *relax* and *gesture of interest*, respectively. Since this array carries information about the sEMG signals from the muscles, we expect each person to have his/her own τ_u. That is, although a given gesture requires the contraction of a specific group of muscles, the contraction strength is a more particular characteristic that may vary between different people.

Fig. 7 depicts the segmentation of two training examples from the same gesture and person using different threshold values. It is worth highlighting that the base value used in [18] was $\tau_u = 10$. Fig. 7(a) illustrates the segmentation result using $\tau_u = 16$ and the raw one-channel sEMG. One may notice that such a threshold value selects the beginning and end of the gesture of interest. However, the wrong choice of τ_u can give us the segmentation in Fig. 7(b), which was generated using the base threshold value $\tau_u = 10$. In this example, the signal did not have segmentation, which resulted in the selection of the entire filtered signal. Therefore, the usage of this threshold value does not result in the extraction of muscle activity.

Fig. 8 shows another example of what may happen when we wrongly choose τ_u. In Fig. 8(a), we perform segmentation using $\tau_u = 16$, and as in the scenario depicted in Fig. 7(a), the processing selects a time range where there is, most likely, only the muscle contraction. On the other hand, we used $\tau_u = 13$ in Fig. 8(b). As a result, the segmentation was faulty. Despite selecting the beginning of the gesture precisely, the segmentation was unable to track the end.

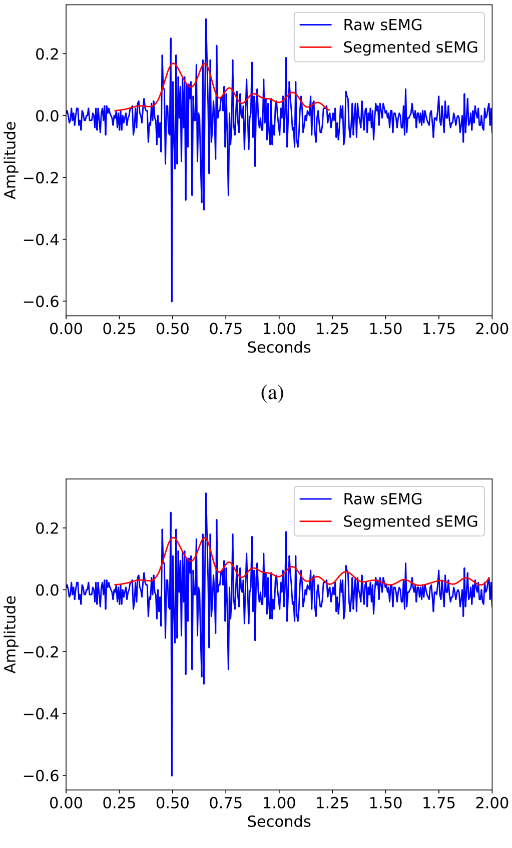


(a)

(b) .

Fig. 7: Segmentation and raw sEMG from the same data example with different threshold values: (a) $\tau_u = 16$; (b) $\tau_u = 10$.

Since there are many threshold values τ_u that can be adequate, we treat the threshold as a hyperparameter, transforming its search into a fine-tuning problem. A common method to solve such a problem is using validation. In this section, we propose to use the k-fold cross-validation method. In such an approach, we divide the training dataset, which is composed of five data examples, into five folds for cross-validation, as depicted in Fig 9. We utilized the k-fold cross-validation for $k = 3, 4, 5$. Fig. 10 illustrates the folds used in each iteration of cross-validation. We used folds 1, 2, and 3 as validation sets for the 3-fold cross-validation. In the 4-fold, we worked with folds 1, 2, 3, and 4. Finally, we used all folds in the 5-fold cross-validation approach. Table II shows the best values for τ_u that we found with the cross-validation method. As expected, the best threshold value changes from subject to subject.



(a)

(b)

Fig. 8: Segmentation and raw sEMG from the same data example with different threshold values: (a) $\tau_u = 16$; (b) $\tau_u = 13$.

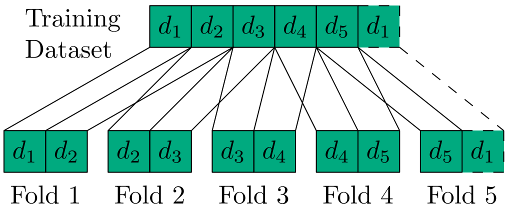


Fig. 9: Training dataset splitting for the cross-validation procedure.

IV. Results and Discussions

In this section, we present the simulation results. We designed some experiments to evaluate the cross-validation method in the current testing dataset. We also compare the performance using validation with the results reported in [18].

The testing dataset $\mathcal{D}_{\text{test}}$ has a total of 10 healthy volunteers. Considering only the testing set, each person recorded 30 examples of each gesture of interest. Due to this and the fact that our aim is to develop individual classifiers, one may consider having 10 testing datasets with 150 examples each. It is worth highlighting that the classification accuracy of the simulations is computed by averaging the outcome of 100 independent trials.

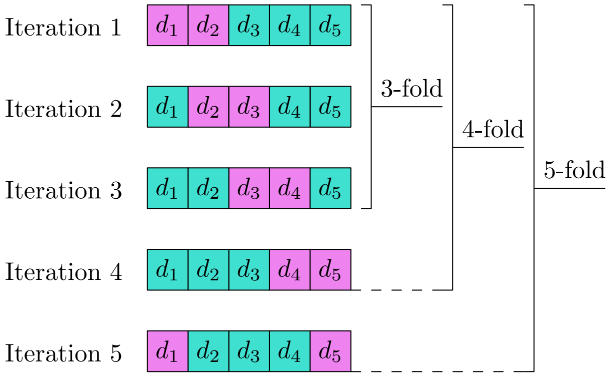


Fig. 10: Training and validation sets for 3-, 4-, and 5-fold cross-validation in each iteration.

TABLE II: Best threshold values τ_u for each subject obtained by cross-validation.

	Subject									
	#1	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**	**#8**	**#9**	**#10**
$\tau_{\mathbf{u,3cross}}$	10	13	12	15	16	10	17	10	13	19
$\tau_{\mathbf{u,4cross}}$	10	13	15	15	17	10	19	10	13	19
$\tau_{\mathbf{u,5cross}}$	10	13	15	15	17	19	19	10	13	19

The baseline classification model has $m_{\text{train}} = 400$ and $m_{\text{test}} = 500$ with a stride between two consecutive time windows equal to 10 points in the data acquisition module. We used a 5th order Butterworth filter with a cutoff frequency of 10 Hz in the pre-processing module. Additionally, the frequency range of the spectrograms is $[0, 100]$ Hz divided into 26 points. In this calculation, we used Hamming windows of length equal to 25 points, with a stride of 15 points. We chose the muscle activity detection threshold as $\tau_u = 10$ for all the subjects. We set the minimum length of segmentation as $a = 100$ points. In the classification module, the constraint for the conditional probability was $\lambda = 0.5$, the regularization weight of the ℓ_2-norm equal to 0.01, and the learning rate of 0.01. We used the hyperbolic tangent function in the hidden layer and the softmax transfer function for the output layer. We trained the FNN with the cross-entropy cost function and gradient descent methods [27], [42]–[44]. The neural network topology used was $6 \times 6 \times 6$. It is worth mentioning that this is the same configuration as in [18].

The datasets we used for this work are part of [18] and can be downloaded at https://drive.google.com/drive/folders/1ZCsaHNc08MYvOS1lfMC_wchioix6srpB, along with the Matlab code for the baseline model. The python code for the proposed model presented in this study can be downloaded at https://github.com/gschaves/gabriel_cbic23.git, as well as the datasets and the parameters setup.

The experiment consists in comparing the mean accuracy classification of the baseline model and the proposed model. The proposed model is the baseline setup but the threshold value, which was acquired by validation. Hence, we evaluate both classification models under the same conditions. We used the same testing datasets for each model to properly compare the results.

We used the 3-, 4-, and 5-fold cross-validation methods described in Section III-B to tune the hyperparameter τ_u. We considered the range of 10 to 20 as the search space. This range of values was obtained by analyzing the entries of $\mathbf{U}$, the array where we extract the indices for segmentation. Another reason we chose such a range was the threshold $\tau_u = 10$ used in the baseline model by the authors in [18]. The proposed classification models used the best threshold values, presented in Table II. It can be seen in Table III that with such values, all models with cross-validation achieved better mean classification accuracy and standard deviation compared to the results of the baseline model. The 3-fold cross-validation method reached $(92.0 \pm 0.8)\%$, an increase of 1.9% in the mean accuracy compared to the baseline. By using 5-fold cross-validation, we achieved 2.9% more mean accuracy than the baseline model. The best classifier was the 4-fold cross-validation, which produced $(93.6 \pm 0.7)\%$ of mean accuracy and standard deviation. On the other hand, the model without gesture detection reached a mean accuracy of 88.1%, representing the worst result among the experimented models.

It is worth reminding that cross-validation is performed in the learning stage. In other words, such a method only interferes with the training process, making the model take more time to be trained. Due to that, its implementation does not increase the system response in the testing. Hence, the proposed model achieves the same classification window processing time reported in [18] of 11 ms.

TABLE III: Mean accuracy and standard deviation results for different classification models, where "no seg" is the model without gesture detection, 3fCV represents the τ_u tuning by the 3-fold cross-validation method, 4fCV is the 4-fold cross-validation, and 5fCV is the 5-fold cross-validation.

Model	**Accuracy (%)**
baseline + no seg	88.1 ± 0.1
baseline	90.1 ± 0.8
baseline + 3fCV	92.0 ± 0.8
baseline + 4fCV	$\mathbf{93.6 \pm 0.7}$
baseline + 5fCV	93.0 ± 0.7

Another interesting result to evaluate is the individual performance of the proposed model using 4-fold cross-validation. In other words, we look at how the system performs when considering each subject separately. Table IV depicts the mean accuracy and standard deviation that we have achieved for each subject model. It is worth highlighting that the results were calculated by averaging the 150 test examples for each person and the outcome of 100 independent trials. The mean accuracy we reached ranges from 78.1% to 99.9%, which shows that the sEMG data can drastically vary from person to person. Even in our case, where each subject has an individually trained model, the sEMG signal patterns may differ due to the unique ways that each person performs a hand movement. Evaluating the model in this manner, we attained a performance of $(93.6 \pm 6.7)\%$.

TABLE IV: Mean accuracy and standard deviation results for each subject separately considering the proposed model (baseline + 4fCV).

Subject	**Accuracy (%)**
#1	97.8 ± 0.7
#2	97.2 ± 0.8
#3	78.1 ± 3.6
#4	96.0 ± 0.2
#5	88.5 ± 4.8
#6	88.0 ± 1.6
#7	92.3 ± 2.7
#8	99.9 ± 0.3
#9	98.8 ± 0.5
#10	99.9 ± 0.1

V. Conclusion

A classification system using ANN and dtw features was able to classify six hand gestures using sEMG data from 10 healthy people as the input. An important part of the system is the segmentation performed by the pre-processing module. In order to reach better results, we used user-specific setups in the segmentation processing for each person. Every model using any validation method attained better results than the baseline configuration and the model without any segmentation. However, the best performance was 93.6%, which we achieved by using 4-fold cross-validation to train individual hyperparameters. Furthermore, we conclude that each person has a unique way of performing a hand movement, as expected.

For future research, we can increase the classifier mean accuracy and standard deviation by using other validation or segmentation techniques since several research works reached great results by combining the sliding window approach with gesture detection.

Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. It was also partially financed by CNPq and FAPERJ.

References

[1] G.-C. Luh, Y.-H. Ma, C.-J. Yen, and H.-A. Lin, "Muscle-gesture robot hand control based on semg signals with wavelet transform features and neural network classifier," in *2016 International Conference on Machine Learning and Cybernetics (ICMLC)*, vol. 2, 2016, pp. 627–632.

[2] N. Nasri, S. Orts-Escolano, and M. Cazorla, "An sEMG-controlled 3D game for rehabilitation therapies: Real-time time hand gesture recognition using deep learning techniques," *Sensors*, vol. 20, no. 22, 2020.

[3] C. L. Toledo-Peral, G. Vega-Martínez, J. A. Mercado-Gutiérrez, G. Rodríguez-Reyes, A. Vera-Hernández, L. Leija-Salas, and J. Gutiérrez-Martínez, "Virtual/augmented reality for rehabilitation applications using electromyography as control/biofeedback: Systematic literature review," *Electronics*, vol. 11, no. 14, p. 2271, 2022.

[4] D. Farina, N. Jiang, H. Rehbaum, A. Holobar, B. Graimann, H. Dietl, and O. C. Aszmann, "The extraction of neural information from the surface emg for the control of upper-limb prostheses: Emerging avenues and challenges," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 4, pp. 797–809, 2014.

[5] Marlis Gonzalez-Fernandez. Amputation: Recovery and Rehabilitation. https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/amputation/amputation-recovery-and-rehabilitation. Accessed: June 02, 2022.

[6] C. Behrend, W. Reizner, J. A. Marchessault, and W. C. Hammert, "Update on advances in upper extremity prosthetics," *The Journal of Hand Surgery*, vol. 36, no. 10, pp. 1711–1717, 2011.

[7] M. E. Huang, C. E. Levy, and J. B. Webster, "Acquired limb deficiencies. 3. prosthetic components, prescriptions, and indications," *Archives of Physical Medicine and Rehabilitation*, vol. 82, no. 3, pp. S17–S24, 2001.

[8] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. Mittaz Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Clinical parameter effect on the capability to control myoelectric robotic prosthetic hands," *Journal of Rehabilitation Research and Development*, vol. 53, no. 3, pp. 345–358, 2016.

[9] A. A. of Orthopaedic Surgeons, *Atlas of Limb Prosthetics: Surgical, Prosthetic, and Rehabilitation Principles*, 2nd ed., ser. Medical, Physical Medicine & Rehabilitation. Maryland Heights, Missouri: Mosby Year Book, 1992.

[10] M. Zanghieri, S. Benatti, A. Burrello, V. Kartsch, F. Conti, and L. Benini, "Robust real-time embedded emg recognition framework using temporal convolutional networks on a multicore iot processor," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 2, pp. 244–256, 2019.

[11] E. A. Clancy, E. L. Morin, and R. Merletti, "Sampling, noise-reduction and amplitude estimation issues in surface electromyography," *Journal of Electromyography and Kinesiology*, vol. 12, no. 1, pp. 1–16, 2002.

[12] W. Li, P. Shi, and H. Yu, "Gesture recognition using surface electromyography and deep learning for prostheses hand: State-of-the-art, challenges, and future," *Frontiers in Neuroscience*, vol. 15, 2021.

[13] D. Farina and O. Aszmann, "Bionic limbs: Clinical reality and academic promises," *Science Translational Medicine*, vol. 6, no. 257, pp. 257ps12–257ps12, 2014.

[14] E. J. Scheme, K. B. Englehart, and B. S. Hudgins, "Selective classification for improved robustness of myoelectric control under nonideal conditions," *IEEE Transactions on Biomedical Engineering*, vol. 58, no. 6, pp. 1698–1705, 2011.

[15] C. Nissler, N. Mouriki, and C. Castellini, "Optical myography: Detecting finger movements by looking at the forearm," *Frontiers in Neurorobotics*, vol. 10, 2016.

[16] S. Amsuess, I. Vujaklija, P. Goebel, A. D. Roche, B. Graimann, O. C. Aszmann, and D. Farina, "Context-dependent upper limb prosthesis control for natural and robust use," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 7, pp. 744–753, 2015.

[17] N. Jiang, H. Rehbaum, I. Vujaklija, B. Graimann, and D. Farina, "Intuitive, online, simultaneous, and proportional myoelectric control over two degrees-of-freedom in upper limb amputees," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 22, no. 3, pp. 501–510, 2014.

[18] M. E. Benalcázar, C. E. Anchundia, J. A. Zea, P. Zambrano, A. G. Jaramillo, and M. Segura, "Real-time hand gesture recognition based on artificial feed-forward neural networks and emg," in *2018 26th European Signal Processing Conference (EUSIPCO)*, 2018, pp. 1492–1496.

[19] A. Jaramillo-Yánez, M. E. Benalcázar, and E. Mena-Maldonado, "Real-time hand gesture recognition using surface electromyography and machine learning: A systematic literature review," *Sensors*, vol. 20, no. 9, 2020.

[20] X. Chen and Z. J. Wang, "Pattern recognition of number gestures based on a wireless surface emg system," *Biomed. Signal Process. Control.*, vol. 8, pp. 184–192, 2013.

[21] W. Shi, Z.-J. Lyu, S.-T. Tang, T.-L. Chia, and C.-Y. Yang, "A bionic hand controlled by hand gesture recognition based on surface emg signals: A preliminary study," *Biocybernetics and Biomedical Engineering*, vol. 38, pp. 126–135, 2018.

[22] J. Yang, J. Pan, and J. Li, "sEMG-based continuous hand gesture recognition using GMM-HMM and threshold model," in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2017, pp. 1509–1514.

[23] C. Yang, J. Long, M. A. Urbin, Y. Feng, G. Song, J. Weng, and Z. Li, "Real-time myocontrol of a human–computer interface by paretic muscles after stroke," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 4, pp. 1126–1132, 2018.

[24] S. Benatti, F. Casamassima, B. Milosevic, E. Farella, P. Schönle, S. Fateh, T. Burger, Q. Huang, and L. Benini, "A versatile embedded platform for emg acquisition and gesture recognition," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, no. 5, pp. 620–630, 2015.

[25] D. V. Redrovan and D. Kim, "Hand gestures recognition using machine learning for control of multiple quadrotors," in *2018 IEEE Sensors Applications Symposium (SAS)*, 2018, pp. 1–6.

[26] U. Côté Allard, F. Nougarou, C. L. Fall, P. Giguère, C. Gosselin, F. Laviolette, and B. Gosselin, "A convolutional neural network for robotic arm guidance using sEMG based frequency-features," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 2464–2470.

[27] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[28] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[29] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE transactions on biomedical engineering*, vol. 40, no. 1, pp. 82–94, 1993.

[30] M. Atzori, A. Gijsberts, C. Castellini, B. Caputo, A.-G. M. Hager, S. Elsig, G. Giatsidis, F. Bassetto, and H. Müller, "Electromyography data for non-invasive naturally-controlled robotic hand prostheses," *Scientific data*, vol. 1, no. 1, pp. 1–13, 2014.

[31] J. Fan, M. Jiang, C. Lin, G. Li, J. Fiaidhi, C. Ma, and W. Wu, "Improving semg-based motion intention recognition for upper-limb amputees using transfer learning," *Neural Computing and Applications*, pp. 1–11, 2021.

[32] S. Thusneyapan and G. Zahalak, "A practical electrode-array myoprocessor for surface electromyography," *IEEE Transactions on Biomedical Engineering*, vol. 36, no. 2, pp. 295–299, 1989.

[33] V. T. Inman, H. Ralston, J. De C.M. Saunders, M. Bertram Feinstein, and E. W. Wright, "Relation of human electromyogram to muscular tension," *Electroencephalography and Clinical Neurophysiology*, vol. 4, no. 2, pp. 187–194, 1952.

[34] J. G. Kreifeldt, "Signal versus noise characteristics of filtered emg used as a control source," *IEEE Transactions on Biomedical Engineering*, vol. BME-18, no. 1, pp. 16–22, 1971.

[35] Y. St-Amant, D. Rancourt, and E. Clancy, "Effect of smoothing window length on rms emg amplitude estimates," in *Proceedings of the IEEE 22nd Annual Northeast Bioengineering Conference*, 1996, pp. 93–94.

[36] M. Müller, *Information Retrieval for Music and Motion*. Berlin, Heidelberg: Springer-Verlag, 2007.

[37] P. S. R. Diniz, E. A. B. da Silva, and S. L. Netto, *Digital Signal Processing: System Analysis and Design*, 2nd ed. Cambridge University Press, 2010.

[38] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed. USA: Prentice Hall Press, 2009.

[39] S. K. Mitra, *Digital Signal Processing: A Computer Based Approach*, 1st ed. USA: McGraw-Hill, Inc., 1997.

[40] M. H. Hayes, *Statistical digital signal processing and modeling*, 1st ed. John Wiley & Sons, 1996.

[41] S. Haykin, *Communication Systems*, 5th ed. Wiley Publishing, 2009.

[42] ——, *Neural Networks: A comprehensive foundation*, 2nd ed. USA: Prentice Hall, 2004.

[43] P. S. R. Diniz, M. L. R. de Campos, W. A. Martins, M. V. S. Lima, and J. A. Apolinário, Jr, *Online Learning and Adaptive Filters*. Cambridge University Press, 2022.

[44] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed. USA: Wiley-Interscience, 2000.