IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 23 July 2024, accepted 19 August 2024, date of publication 22 August 2024, date of current version 2 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3448410

RESEARCH ARTICLE

sEMG-Based Gesture Classifier Through DTW and Enhanced Muscle Activity Detection

GABRIEL S. CHAVES, (Member, IEEE), ANDERSON S. VIEIRA, AND MARKUS V. S. LIMA, (Member, IEEE)
Electrical Engineering Program (PEE/Coppe), Universidade Federal do Rio de Janeiro (UFRJ), Rio de Janeiro 21941-909, Brazil
Corresponding author: Gabriel S. Chaves (gabriel.chaves@smt.ufrj.br)

This work was supported in part by Coordenação de Aperfeiçoamento de Pessoal de Nível Superior—Brasil (CAPES) under Finance Code 001, and in part by the Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) and Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ).

ABSTRACT Prosthetic hands are of paramount importance in the rehabilitation of upper limbs amputees. Gesture recognition using surface electromyography (sEMG) data has emerged as a great option for controlling prosthetic devices, since these data are acquired by non-intrusive sensors. This work presents a real-time classification system based on artificial neural networks with individualized muscle activity segmentation and using dynamic time warping (DTW) based features. For real-time classification, we modify the size of the sliding windows so that their length is sufficient to fully capture the muscle activity signal. For data segmentation, we propose an enhanced muscle activity detection in which validation is used to fine-tune the thresholds needed to determine the beginning and end of muscle contraction. We used two validation methods: cross-validation and multi-holdout. Moreover, we propose a post-processing technique to choose the most representative class when there are multiple classifications for a given data. By combining all the proposed techniques, the accuracy of the resulting system was $(\mathbf{97.2 \pm 0.3})\%$ in the classification of $\mathbf{6}$ hand gestures from $\mathbf{10}$ healthy people, representing an increase of $\mathbf{7.1}\%$ in the mean accuracy compared with the baseline model.

INDEX TERMS Hand gesture classification, muscle-activity detection, signal processing, sEMG, DTW, neural network.

I. INTRODUCTION

Losing a limb can be traumatic, having an enormous impact on a person's body, emotions, relationships, vocation, and way of life. While some other surgical procedures return the patient's health relatively quickly, the recovery period after a major limb amputation can take a long time [1]. For patients facing an amputation, rehabilitation can lead to the use of prostheses. Unfortunately, the lack of usability of upper limb prostheses contributes to its high rejection rate [2], [3]. Hence, in the last decades, the human-machine interface field has brought many new applications to motor rehabilitation, such as the control of myoelectric hand prostheses, aiming to decrease this rejection rate [4], [5]. Myoelectric devices are controlled by *electromyography* (EMG) signals generated by muscle contractions, and the problem of hand gesture recognition consists in classifying these muscle contractions [6].

Currently, non-intrusive myoelectric devices/sensors that perform *surface EMG* (sEMG) are commonly used. An sEMG signal is the EMG reading over the skin surface, and it captures the spatio-temporal interference pattern of the electrical activity of the motor neurons [7]. This electrical pulse, which is the response of the motor neuron to the brain's command to make a movement [7], [8], [9], [10], can be used to drive an external device, like a prosthesis. In addition to being non-intrusive, a major advantage of sEMG sensors is their easiness of operation, which allows the existence of many simple and reliable commercial devices [11], [12], [13].

Regarding the gesture classification problem, its solution aims to identify muscle contraction patterns in the sEMG signal and match them with pre-defined gestures through supervised learning methods [14]. Advances in machine

The associate editor coordinating the review of this manuscript and approving it for publication was Li Zhang.

learning made possible the development of several techniques in pattern recognition applications. Several research works have been achieving great results by using classical machine learning techniques such as the K-Nearest Neighbors (K-NN) [15], Support Vector Machine (SVM) [16], and Linear Discriminant Analysis (LDA) [17]. On the other hand, artificial neural networks (ANNs) reached the state-of-the-art in many research fields, such as image recognition [18], speech recognition [19], and natural language processing [20], [21], [22]. Additionally, approaches using ANNs have been reaching results above 80% in classifying several hand gestures [23], [24]. For example, in [25], the authors presented a classifier that recognized 5 hand gestures with an accuracy of 90.7%, using an ANN with time-domain features. The authors in [26] used an ANN that obtained 95% accuracy when classifying 8 hand gestures. The combination of time-frequency domain features with an ANN having 3 layers achieved an accuracy of 93.25% in [27], classifying 3 hand and wrist movements of four subjects. On a more general case, the authors in [28] proposed a recognition system able to classify 10 hand gestures, achieving 94% mean accuracy with four subjects, using a time-delayed ANN. In [29], the authors reported an average recognition accuracy of 85.08% using a complex ANN structure to extract features and classify 5 hand gestures. It is worth highlighting that the results of these research works are not comparable since they use different databases. That is, no fair comparison among these results/works can be made.

The *general structure of the classification system* adopted in this work was proposed in [23]. It consists of the following modules: (i) data reading; (ii) pre-processing; (iii) feature extraction and selection; (iv) classification; and (v) post-processing. The first module reads the data examples from the datasets, being responsible for the system input in the training and testing stages. In our case, the data reading module inputs the data signals through a sliding window method, which alters the dimensionality and sometimes the form of the signal itself [7]. The pre-processing module extracts the hand movement data from the sEMG signal through muscle activity detection. There are many features that can be extracted from an sEMG signal, such as time- and frequency-domain features. The feature extraction/selection module selects a proper set of features. The classification module is where the ANN operates by matching selected feature patterns to their respective labels. Finally, the post-processing module refines the classification output, which helps to interpret and evaluate the model results. This is an important step since the sliding window procedure splits each data example into several observation windows, thus yielding several results that must be processed before presenting the final classification result to the user.

Although we use the same general structure as in [23], we propose modifications in some modules that compose the classification system. In addition to that, we present an extensive study on some peculiarities of the system, which one needs to be careful when working on it. The *main contributions* of this work are summarized as follows:

1) IN DATA READING MODULE

We modify the window length. The reasoning is that by increasing the window length during the test stage, the data structure during this stage becomes more similar to the one of the training stage, thus improving the classification, as it will be clear in the next sections.

2) IN PRE-PROCESSING MODULE

We propose an enhanced muscle activity detection methodology to segment the sEMG signals more accurately. This methodology consists in applying model validation techniques in order to tune a key hyperparameter for each person, instead of having a single parameter for all individuals like in [23]. Hence, in our approach, the system is capable of automatically adapting itself to each person.

3) IN CLASSIFICATION MODULE

We apply an overcomplete ANN. Overcomplete neural networks have the potential to capture intricate patterns in the data [30]. Thus, in the proposed methodology, we increase the number of neurons in the hidden layer and compare the different resulting architectures.

4) IN POST-PROCESSING MODULE

We propose a simple voting system, herein called *poll*, to select the most representative classification result among all the results obtained for the different windows belonging to a given sEMG example.

In summary, in this work, we take the hand gesture classification (HGC) system proposed in [23] as our starting point (*baseline model*) and *we propose modifications in 4 out of its 5 modules* (see items 1 to 4 above for a brief description). These modifications are not only tested using real data but they are also discussed so that their theoretical rationale is justified. Moreover, each of the proposed contributions increased the mean accuracy of the classification system in comparison to the baseline model, thus corroborating our arguments. Besides, when we combined all the proposed modifications, we obtained the *best setup* for the HGC system, achieving 97.2% in the mean accuracy, thus surpassing the baseline model's mean accuracy by 7.1%. Also important is the fact that such an increase in mean accuracy is obtained without increasing the processing time during the test stage, aka *inference time*, meaning that the resulting HGC system meets real-time requirements just like the baseline model does [23]. All codes and data necessary to reproduce our results are publicly available (we provide the links at the beginning of Section V).

This work is organized as follows. Section II details the datasets we used. The HGC system is described in Section III. In Section IV, we introduce the proposed validation methodologies for hyperparameter tuning. In Section V, we present experiments and results. There is a careful discussion of

results in Section VI. Finally, the conclusion is drawn in Section VII.

II. DATASETS

In this work, we used the public datasets introduced in [23] and, for the sake of clarity, we provide a brief description of them. The datasets are comprised of sEMG signals that were recorded using the Myo armband, a commercial low-cost EMG sensor built by Thalmic Labs. By wearing this device on the forearm, the sensors can return the digital sEMG signal from the flexor, pronator, and extensor muscles to a computer via Bluetooth [31]. The Myo armband resulting data is comprised of the sEMG reading of 8 channels operating at a sampling rate of 200 Hz. Each channel outputs the measurement of a sensor that composes the armband. The sensors are not placed over specific muscles. Instead, they are distributed aiming to cover the surface of the forearm.

The datasets are comprised of sEMG signals from 6 hand gestures. There are 5 *gestures of interest*, namely: 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 the focus of our work, we use many of its characteristics to help distinguish the several sEMG patterns of interest. Additionally, we use the label no-gesture (*no*) for all the gestures our model cannot classify or the ones that are not of interest, including the rest position. Figure 1 illustrates the gestures of interest in their final hand positions.

FIGURE 1. Final hand position for the 5 gestures of interest.

There are two datasets, one for training and the other for testing. They are composed of signals/data acquired from 10 healthy volunteers. Each data example from both datasets is comprised of an 8-channel sEMG signal consisting of continuous hand movements in the following sequence: rest position, gesture of interest, and rest again. Additionally, each example has a single label corresponding to the gesture of interest, but the time instants in which the gesture begins and ends are not provided by the datasets. The knowledge of the true label of the data examples enables the application of supervised learning techniques.

Figure 2 depicts one of the channels of an sEMG example. In this figure, one can observe the temporal patterns that exist in the recordings. We can divide an sEMG signal into three main parts. In the first part of the data, labeled as 1 in the left side of the figure, one observes a low-magnitude sEMG signal corresponding to the rest position. In the second part, labeled as 4 in the figure, we have the gesture of interest. In this part, the sEMG magnitudes increase, as some muscles are active. In the third portion, labeled as 1 in the right side of the figure, the magnitudes decrease indicating that the hand has returned to the rest position. Clearly, the second portion of the data is the one that matters to recognize the gesture of interest, and can be considered a random process comprised of two components: the transient and the steady-state [7], [8], [23], [32], [33].

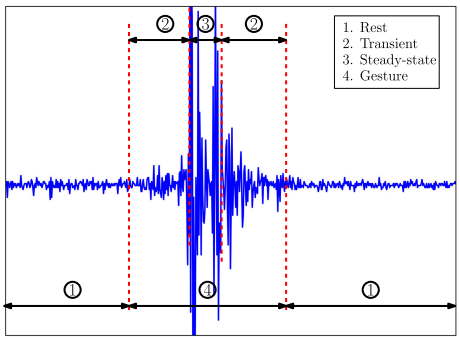

FIGURE 2. General structure of a one-channel sEMG example.

1) TRAINING DATASET

Since one of the main ideas proposed in this work is to design a particular machine learning model for each person, we can consider the training dataset as 10 individual sets. The training dataset $\mathcal{D}_{\text{train}} = \{(\underline{\mathbf{F}}_1, Y_1), \ \ldots \ , (\underline{\mathbf{F}}_N, Y_N)\}$ of each user 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 sEMG signal measured. Each $\underline{\mathbf{F}}_i$ has a duration of approximately 2 seconds (400 samples). 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 1 shows the relation between Y_i and the corresponding gestures. Each one of the classes that compose Y_i has 5 examples, totaling all 30 training examples for each user. As previously described, the volunteers were asked to perform hand movements to acquire the respective data for each class. When recording the no-gesture examples, the users kept their hands in the rest position during the whole measurement time of 2 seconds.

TABLE 1. 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*)

2) TESTING DATASET

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 consists of $M = 150$ examples recorded for 5 seconds each. It is worth mentioning that the class no-gesture was not included in $\mathcal{D}_{\text{test}}$ since our goal is to classify the 5 gestures of interest. Thus, the testing set $\mathcal{D}_{\text{test}}$ has 30 examples of each class of $\hat{Y}_i \in \{2, 3, 4, 5, 6\}$, totaling 150 testing examples for each user. In the testing

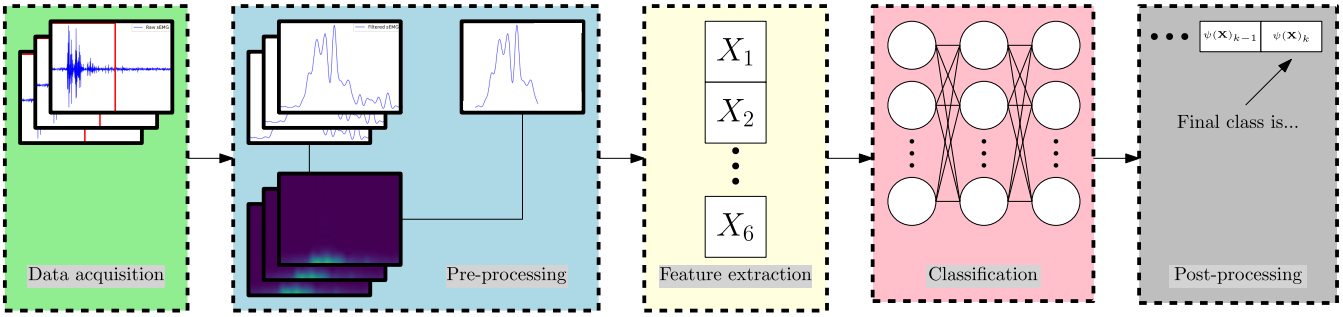

FIGURE 3. Framework illustrating the 5 modules of the proposed HGC system.

set, the 8-channel sEMG signal is $\underline{\mathbf{G}}_i \in [-1, 1]^{1000\times 8}$, with $i = 1, 2, \ldots, M$.

III. THE HGC SYSTEM

This section introduces the HGC system used to identify the gestures described in Section II. As depicted in Figure 3, this system can be divided into five modules: data acquisition, pre-processing, feature extraction/selection, classification, and post-processing. In this section, we discuss each of these modules, explaining in detail their main tasks and purposes.

1) DATA ACQUISITION

The data acquisition module is responsible for reading the 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 $i = 1, 2, \ldots, m$ [7], [34], [35], [36], [37], [38]. In this work, W is a rectangular window [39]. In the training stage, we set the length m_{train} of the window W as $m_{\text{train}} = 400$ points, i.e., a time window that selects all points of each training example. We chose this approach since the training examples have only a single label, despite having multiple gestures in the data (like the *rest* and gesture of interest, see Figure 2). In the testing stage, one may choose m_{test} according to the computational constraints and processing time. Additionally, the stride between two consecutive time windows is 10 points, ensuring that the data collection process occurs rapidly (in this case, in 50 ms). 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 $i = 1, 2, \ldots, 8$.

2) PRE-PROCESSING

The main objective of the pre-processing module is to detect and extract the portion of the sEMG signal containing the muscle activity. The first step consists in rectifying the signal $\underline{\mathbf{E}}$, which results in the new signal $\underline{\mathbf{R}} = \text{abs}(\underline{\mathbf{E}}) \in [0, 1]^{m\times 8}$, i.e., we take the absolute value of all the entries of $\underline{\mathbf{E}}$. In the second step, we apply a Butterworth low-pass filter Φ with a cutoff frequency of 10 Hz to each channel of $\underline{\mathbf{R}}$ [39], [40], [41], [42]. This process yields the signal $\underline{\mathbf{V}} = \Phi(\underline{\mathbf{R}}) \in [0, 1]^{m\times 8}$, an envelope of $\underline{\mathbf{R}}$. The third step is responsible for the reduction of the signal dimension. We accomplish this by summing along the channel axis of $\underline{\mathbf{V}}$, resulting in $\mathbf{S} = \text{sum}(\underline{\mathbf{V}}) \in [0, 8]^{m\times 1}$. In the fourth step, 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 $\mathbf{S}$ by dividing the 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 $\underline{\mathbf{V}}$ [43]. The fifth step calculates the matrix $\underline{\mathbf{P}} \in \mathbb{R}^{26\times p}$, in which each entry is the modulus of $\underline{\mathbf{P}}_C$. In the next step, we reduce the dimension of matrix $\underline{\mathbf{P}}$ by summing its rows. Thus, we obtain the vector $\mathbf{U} = \text{sum}(\underline{\mathbf{P}}) \in \mathbb{R}^{1\times p}$. In the last step, we start by finding the indices $i_{u,s}$ and $i_{u,e}$ of $\mathbf{U}$. These indices are the first two positions in which the difference between two consecutive entries of $\mathbf{U}$ is equal to or greater than a given threshold $\tau_u \in \mathbb{R}_+$. Thus, we convert the STFT indices $i_{u,s}$ and $i_{u,e}$ into the sample indices i_s and i_e, where i_s and i_e are the first sample indices of the STFT indices $i_{u,s}$ and $i_{u,e}$, respectively. Furthermore, we need to check if the number of points between i_s and i_e is suitable for the application. That is, if the difference $i_e - i_s$ is less than a given number of points $a \in \mathbb{Z}_+$, then the segmentation returns the signal $\underline{\mathbf{Z}} = \underline{\mathbf{V}}$. Otherwise, we segment $\underline{\mathbf{V}}$, obtaining $\underline{\mathbf{Z}} = \underline{\mathbf{V}}(i_s : i_e, :)$. It is worth highlighting that this segmentation process results in signals $\underline{\mathbf{Z}}$ that may have different lengths. Next, if $i_s = 1$ and $i_e = m$, we return the label no-gesture and proceed to the post-processing module directly. Otherwise, we send the resulting signal $\underline{\mathbf{Z}}$ to the next module.

3) FEATURE EXTRACTION AND SELECTION

The sEMG is a spatio-temporal interference pattern of the muscular electrical activity near the detection surface. Additionally, considering that the muscle activity corresponding to a given gesture can be slow or fast, the related sEMG signal can be long or short, respectively. Hence, we need a feature capable of synthesizing a large amount of temporal and spatial data. Also, we need features that address signals that may vary in length. *Dynamic Time Warping* (DTW) meets these requirements. It 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 [23], [44]. In this module, we first calculate the DTW between all the pre-processed training examples that

have the same labels. This approach generates the DTW distances between the training examples within the same class. Next, we choose the pre-processed signal $\underline{\mathbf{H}}_i^*, \forall i \in \{1, \ldots, 6\}$, that is closest to all elements in each class. In other words, $\underline{\mathbf{H}}_i^*$ is the signal yielding the smaller total DTW distance to the other data examples within the ith class (i.e., it is the most representative example of that class). Finally, the module outputs the feature vector for the signal $\underline{\mathbf{Z}}$ as $\mathbf{X} = (\text{dtw}(\underline{\mathbf{H}}_1^*, \underline{\mathbf{Z}}), \ \ldots \ , \text{dtw}(\underline{\mathbf{H}}_6^*, \underline{\mathbf{Z}}))$, where dtw denotes the DTW distance for multi-channel time series. In summary, the feature vector $\mathbf{X}$ is the DTW between the segmented signal $\underline{\mathbf{Z}}$ outputted by the pre-processing module and the training class centers $\underline{\mathbf{H}}_i^*$.

4) CLASSIFICATION

The classification module is responsible for predicting the gesture Y for a given feature vector $\mathbf{X}$. One can write the predictive modeling problem as

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

subject to the constraint that the conditional probability that maximizes (1) is equal to or greater than λ. In other words, the feature vector $\mathbf{X}$ must have λ or greater probability to correspond to Y. Otherwise, $\mathbf{X}$ receives the no-gesture label. There are several methods that can solve the problem in (1). In this work, we choose a feedforward neural network (FNN). The FNN is one of the simplest neural networks proposed, in which the data flows only in the forward direction [45]. We opt for a shallow FNN having only three layers: input, hidden, and output. It is worth mentioning that the input layer receives the signal $\mathbf{X}$ in its standardized version, calculated by $(X_j - \mu)/\sigma$, where μ and σ denote the mean and standard deviation of $\mathbf{X}$, respectively, with $j = 1, 2, \ \ldots \ , 6$.

5) POST-PROCESSING

This last module of the system is responsible for processing the results before presenting them to the user. By using a sliding window approach to read the data examples from the testing dataset, each example is divided into several time windows. For each window, a label estimate is generated by the classification module. Since each example has the *relax/rest* and gesture of interest hand movements, one would expect to have only two different classifications for each data example. However, this is not quite the case. Some of the windows may be misclassified, ending up with several different classifications for a given example. The post-processing module addresses this issue. In this module, we select one label estimate, among the several estimates corresponding to the time windows, to represent the data example, and then we compare it with the true label of this example in order to evaluate the model's performance.

IV. VALIDATION METHODOLOGIES

Validation is a method to decrease model overfitting [46]. Similar to the test set, the validation set also consists of data that the learning process has not seen. However, unlike the test set, we use the validation set to fine-tune the several parameters that need to be chosen before the training stage. These parameters, known as hyperparameters, are responsible for defining the machine learning model to be trained. As illustrated in Figure 4, the validation set is usually a subset of the training dataset $\mathcal{D}_{\text{train}}$, which we split into two new sets, $\mathcal{D}'_{\text{train}}$ and $\mathcal{D}_{\text{val}}$, for training and validation, respectively.

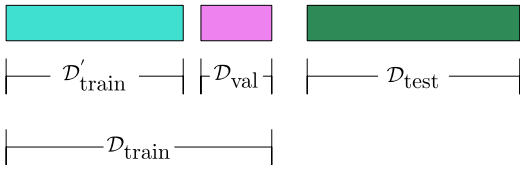

FIGURE 4. Common database split for validation.

1) CROSS-VALIDATION

Usually, we have limited data at our disposal, and in order to design good models, we would like to use as much data as possible for the training. However, a small validation set may lead to an unreliable estimate of the performance due to the use of only a few data points. One solution to this issue is the cross-validation method. In this technique, we separate the dataset $\mathcal{D}_{train}$ into K folds, so that $K - 1$ folds are used to train the models, and we evaluate the performance in the remaining fold (known as validation fold). This process represents one run of the method. We repeat this procedure for all the choices one can make for the *validation fold*, and the final score is computed by averaging the performances from the K runs. It is worth highlighting that such folds can have different sizes [47].

The $K-$fold cross-validation method is great for selecting suitable hyperparameters. Suppose we have L sets of hyperparameters $g_1, \ g_2, \ g_3, \ \ldots \ , \ g_L$, each one is composed by different combinations of the values of λ and τ_u. One may use cross-validation to compare the performance of the models for each combination and choose the one with the lowest cross-validation error. The main drawback of this method is that the number of training runs increases by K. Additionally, this can be much worse depending on the number of hyperparameters that need to be tuned. Combinations of such hyperparameters may require many training runs that are exponential in the number of hyperparameters [47].

2) MULTI-HOLDOUT

In our current database, the $K-$fold cross-validation technique may not yield a good prediction of the test error. As previously described, we work with two distinct datasets, one for training and the other for testing. The main difference between them is that the record time of the data examples is two seconds for the former and five seconds for the latter. Since the training data are different from the testing ones, we could use testing data examples for validation in order to reach better performance. However,

a cross-validation method requires the training and validation data to have exactly the same dimensions and lengths. One must remember that the training data in one iteration of cross-validation is the validation data in another. Hence, in our case, we must use only the data from the training set in cross-validation, which may give us an inaccurate prediction of the test error. An alternative is to use the holdout validation method.

The holdout technique consists of splitting the data into two fixed subsets called a training set and a holdout set. Such a method maintains the concepts of validation, where the model learns in the training set and is validated in the holdout set. Usually, this approach yields a pessimistic approximation of the estimator since only a portion of the data is used for training [48]. However, in our database, we can use the complete training dataset to train the model and select a partition of the test dataset as the holdout set. Thus, validating the trained models with more representative examples for the testing data. It is worth mentioning that such an approach has the disadvantage of evaluating the final model with fewer testing data examples. Figure 5 illustrates the proposed division of our database when applying the holdout method.

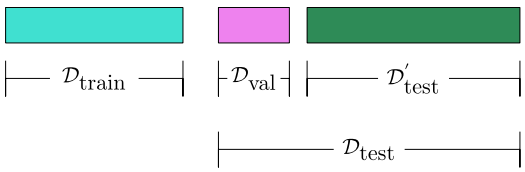

FIGURE 5. Database split for the proposed holdout method.

The holdout method also gives us the best hyperparameters that achieve the lowest validation error. As validation can be used as a good estimation for the prediction error in generalization, such a technique potentially yields the best predictive model [46]. However, due to the stochastic nature of the learning process and random initialization, the network may converge to local minima, sometimes resulting in poor performance. Hence, we propose the Multi-holdout method, where we initiate multiple instances of the holdout method in parallel. Each instance has the same hyperparameter settings except for the initialization weights. In other words, each one starts from a different point. After training and validation in the holdout set, we select the model from the instance with the lowest validation error. This approach reduces the variance of the holdout technique. The high variance in the prediction is one of the major disadvantages of this method since, unlike cross-validation, it does not take the average of multiple runs to output its results.

V. EXPERIMENTAL RESULTS

The experiments were designed to verify the performance of the classification system when we: (i) have more similar training and testing data by increasing the length of the time window in the testing stage; (ii) apply a new post-processing method; (iii) train a unique τ_u for each subject combined with a validation methodology; and (iv) use an overcomplete neural network in the classification module. We also present a comparison of our results with a baseline model. The datasets used in this work are part of [23] 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/gesture_rec_NSRE, as well as the datasets and the parameters setup.

A. BASELINE CONFIGURATION

We used the same configuration presented in [23] as the baseline model. Thus, for the baseline, we have $m_{\text{train}} = 400$ and $m_{\text{test}} = 500$ with a stride between two consecutive time windows equal to 10 points. 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 10 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 was equal to 0.01, and the learning rate was 0.01. We used the hyperbolic tangent function in the hidden layer and the softmax activation function for the output layer. We trained the FNN with the cross-entropy cost function and the gradient descent optimization method [45], [47], [49]. The neural network topology was comprised of 6 inputs, 6 neurons in the hidden layer, and 6 outputs. We computed all the results by averaging the outcome of 100 independent trials.

B. DATA ACQUISITION WINDOW LENGTH

Since we train the system with features considering data having complete gestures, we must test with data having this same characteristic. We can achieve this by adjusting the length of the testing observation windows. Figs. 6 and 7 depict data readings with a time window at the beginning of the example and another at the end. However, Figure 6 shows a time window of length $m_{\text{test}} = 500$, whereas in Figure 7 we have $m_{\text{test}} = 700$. In Figure 6, we can notice that the window length is not large enough to read the complete gesture. Figure 6a presents the first window reading, and the signal inside it does not have the complete muscle activity data from the example. The same happens in Figure 6b, where the beginning of the gesture is missing. In both cases, classification errors are very likely to occur. On the other hand, the window length $m_{\text{test}} = 700$ appears to be more adequate for this example, as illustrated in Figure 7. In Figs. 7a and 7b, the complete gesture seems to be inside each window.

We can not neglect the window length in the classification system [7], [38]. Hence, we ran an experiment in order to

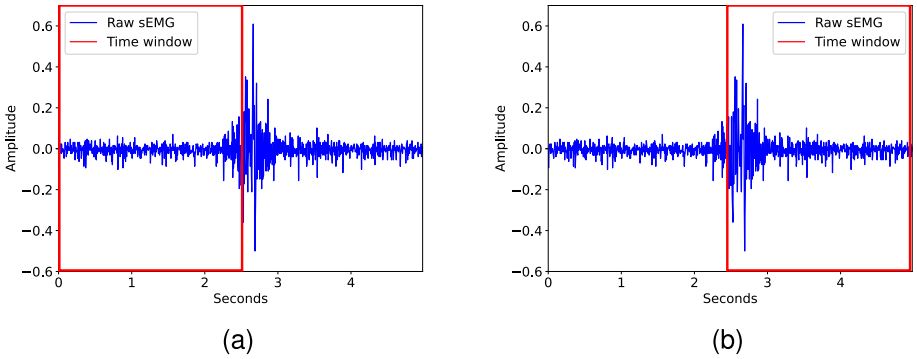

FIGURE 6. Windows of length $m_{\text{test}} = 500$ capturing an incomplete gesture in a test data example: (a) First time window; (b) Last time window.

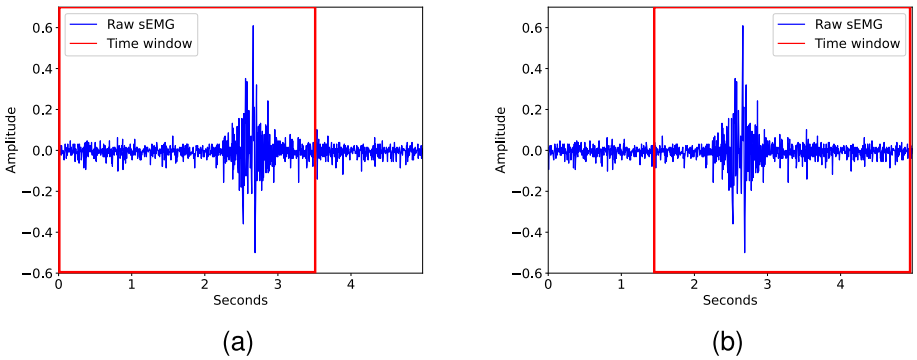

FIGURE 7. Windows of length $m_{\text{test}} = 700$ capturing a complete gesture in a test data example: (a) First time window; (b) Last time window.

evaluate the impact of increasing the window length m_{test} to 700. Table 2 shows that almost all models with $m_{\text{test}} = 700$ had better mean accuracy than those with $m_{\text{test}} = 500$. For instance, by changing only the window length in the baseline setup, the classifier with $m_{\text{test}} = 700$ outperformed the mean accuracy of the baseline model by 1.4%. It is worth highlighting that the increase in the window length does not imply more latency to collect the data, as the stride between consecutive windows is equal to 10 samples for all models. The only exception occurs during the data buffering for the first window, where the larger window requires more time to collect the data; therefore, such delay is irrelevant in the long run.

TABLE 2. Mean accuracy and standard deviation results for different models, where 3fCV represents the τ_u tuned via 3-fold cross-validation, $m700$ indicates that the window length was $m_{\text{test}} \triangleq 700$, *poll* is the polling method in the post-processing. All models used a 6 × 6 × 6 neural network.

Model	Accuracy (%) (Mean)±(1 std)
baseline [23]	90.1 ± 0.8
baseline + $m700$	91.5 ± 0.6
baseline + 3fCV	93.1 ± 0.7
baseline + $m700$ + 3fCV	96.1 ± 0.4
baseline + *poll*	92.3 ± 0.5
baseline + $m700$ + *poll*	91.8 ± 0.5
baseline + 3fCV + *poll*	94.9 ± 0.5
baseline + m700 + 3fCV + poll	$\mathbf{96.5 \pm 0.4}$

C. POST-PROCESSING LABEL SELECTION

As described in Section II, each example has sEMG data representing the following hand movements: rest, the gesture of interest, and rest. However, only the label for the gesture of interest is present in each data example. In other words, even though the system was designed to work in real-time by classifying short time windows, we still need to wait for the full example in order to create the desired metric. Additionally, we must decide on a method to select the most representative gesture classified in each window as the example class. Hence, we can evaluate the models' performance by comparing the prediction with the true label from the dataset.

There are several methods to decide which prediction to pick among those classified in the time windows, i.e., the predicted gesture to be the final classification for a given data example. Suppose we are analyzing the output of the classification module, after the system classifies the several time windows that compose the input data example. The authors in [23] used a simple approach that can select one of these predictions as the final classification. The first step of the method consists of removing the *rest* classifications since the focus is on the other gestures. In the second step, they sort the labels in ascending order, according to Table 1. Lastly, they choose the first sorted label as the representative

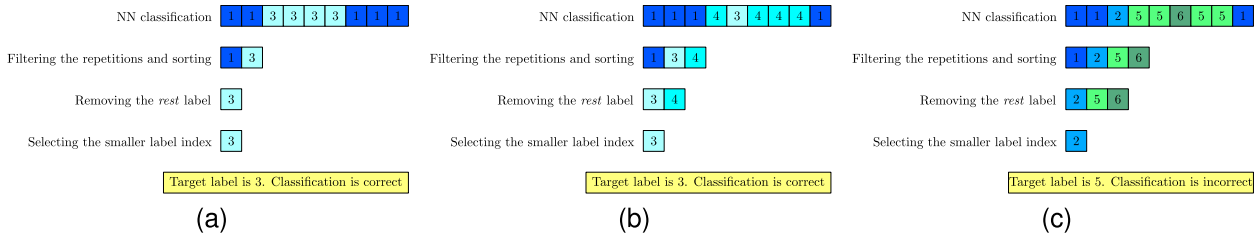

FIGURE 8. Exemplifications of the post-processing methodology implemented in the reference paper [23]: (a) Correct classification when the network outputs only one label as the gesture of interest; (b) Correct classification by "luck" when the network outputs two labels as the gestures of interest; (c) Incorrect classification when the network outputs three labels as the gestures of interest.

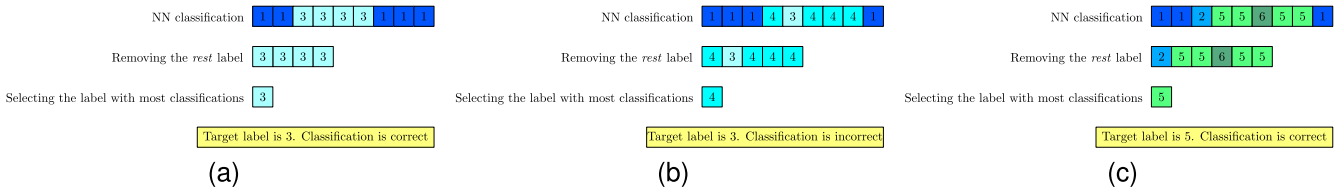

FIGURE 9. Exemplifications of the proposed post-processing methodology: (a) Correct classification when the network outputs only one label as the gesture of interest; (b) Incorrect classification when the network outputs two labels as the gestures of interest and most of the classifications are different from the true label; (c) Correct classification when the network outputs three labels as the gestures of interest and most of the classifications correspond to the true label.

for that data example. This technique performs very well when all the time windows are classified with the same label, which is likely the correct one. However, a single time window misclassification can result in choosing the wrong label for the example.

Figure 8 illustrates several cases using such a post-processing method. Figure 8a exemplifies a case where the example data target/true label is 3. The data was split into 9 time windows, outputting the following vector of classifications: 1, 1, 3, 3, 3, 3, 1, 1, 1. After filtering and sorting, this post-processing approach returns the label 3 as the data example. In this case, the label was classified correctly, as expected. Since the time windows classifications output only one gesture of interest and, in this case, the correct one, such an approach will positively respond. Figure 8b also depicts an exemplification for a data example with a target label equal to 3. In this case, the classification sequence given by the neural network was 1, 1, 1, 4, 3, 4, 4, 4, 1. Because of the filtering, sorting, and removal of the label 1, the method outputs labels 3. In this exemplification, this technique also classifies the data correctly. However, it is a misleading result. For example, if such a scenario occurs in the training stage, the neural network is not learning to classify the label 3. Since most of the classifications point to the label 4 and the final result is 3, it is most likely that, after convergence, the network will mistake data that has the label 3 with those having 4. Figure 8c shows a data target labeled with 5 that was wrongly classified. Despite the neural network classifications being 1, 1, 2, 5, 5, 6, 5, 5, 1, the post-processing method results in the label 2. Such a case exemplifies that even when the network points to the correct label (most time windows have the correct prediction), a single misclassification (label 2) can influence an incorrect result.

In this work, we propose a new approach to address the final classification problem. Unlike the previous method where the label with a smaller index is selected as the final classification for the data example, the new post-processing module (called *poll*) selects the most frequently predicted class across all windows. That is, we implement a voting system in which we output the label corresponding to the majority of the classifications. This method performs well when all the time windows have the same label, like the technique in [23]. In addition, time windows with erroneous classifications have less impact when determining the example label, considering the system classifies the correct one most of the time. Figure 9 depicts the same scenarios presented in Figure 8. The difference is that in Figure 9, we applied our proposed post-processing methodology to select the example label. Figure 9a exemplifies the simplest case, where the sequence classified by the network has only one gesture of interest. The true label for this data example is 3, and the sequence is 1, 1, 3, 3, 3, 3, 1, 1, 1. The first step is to remove the *rest* label, resulting in 3, 3, 3, 3, 3. In the next step, we select the label having the most classifications, which is the label 3. Thus, the classification is correct. The second scenario is where the network outputs two gestures of interest, and the most classified is the incorrect one, as depicted in Figure 9b. Unlike the post-processing performed in [23], our method misclassifies the gesture in this exemplification. The classification sequence of the time windows is 1, 1, 1, 4, 3, 4, 4, 4, 1. By following the proposed technique, we have the new sequence 4, 3, 4, 4, 4, and the final selection is the label 4 since four time windows were

classified with this label, in comparison to the label 3, which had only one classification score. In this case, the true label is 3, hence, an incorrect classification. We have some advantages when comparing this result with the one we obtained using the former post-processing method. By taking the same learning scenario presented previously, when such a classification occurs in the training stage, our proposed post-processing will try to learn how to correctly classify labels 3 since the current path is leading to an erroneous classification (label 4 instead of 3). Figure 9c shows a scenario demonstrating the robustness of our method against certain misclassifications. In this case, the sequence of classifications is 1, 1, 2, 5, 5, 6, 5, 5, 1, i.e., three gestures of interest. The neural network classifies most time windows with the label 5, resulting in the selected label for this data example. The true label of this exemplification is 5, representing that the classification is correct. The proposed post-processing chooses the correct label as the representative for the data example despite some time windows having incorrect classifications (there was one with the label 2 and another with the label 6).

The proposed post-processing module is an easy fix to classification errors when some windows are misclassified. We ran an experiment where we changed the baseline post-processing module into the proposed one and compared the system's performance. Table 2 illustrates that the model using the proposed post-processing method achieved $(92.3 \pm 0.5)\%$ of mean accuracy, 2.2% higher than the baseline model.

D. MUSCLE ACTIVITY DETECTION THRESHOLD

In Section III, one can recall that the signal resulting from the spectrogram calculation is the vector $\mathbf{U}$. This signal is an array composed of low- and high-magnitude values. Ideally, they refer to the *rest* and other gestures, respectively. Thus, we can extract the gesture of interest by comparing these magnitude values to the threshold τ_u, completing the segmentation procedure [23]. Since the entries in $\mathbf{U}$ are relative to the sEMG of each person, we expect each person to have its own threshold value. Indeed, the sEMG carries individual information about each person, such as the power and intensity of hand movements. To illustrate this, suppose a scenario where two people are closing their hands. The first person closes the hand very slowly and employs little force to the movement. The second one also performs the gesture slowly, however employing more force to complete the movement, intensifying the gesture. Such a difference in the execution of the movement may produce different sEMG patterns. In other words, each person will require individual thresholds τ_u in order to segment the gesture, even when they are performing the same hand movement.

We propose two validation methods to choose the best threshold value τ_u. The first consists of the 3-fold cross-validation method. In such an approach, we divide the training dataset, which is composed of 5 data examples, into five folds for cross-validation, as depicted in Figure 10. In this

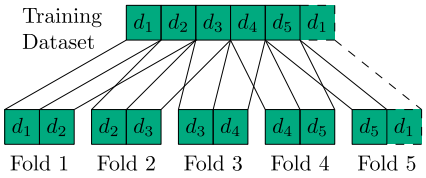

FIGURE 10. Training dataset splitting for the cross-validation procedure.

figure, d_i for $i = 1, \ldots, 5$ represents the training data examples. We used folds 1, 2, and 3 as the validation sets in each iteration of the 3-fold cross-validation method, as one can see in Figure 11. We also used the base threshold value of $\tau_u = 10$ as a hint to create the search space, hence we vary this hyperparameter from 10 up to 20 in searching for the best value. Table 3 shows the best values that we found with the 3-fold cross-validation method. As expected, the best threshold value changes from subject to subject.

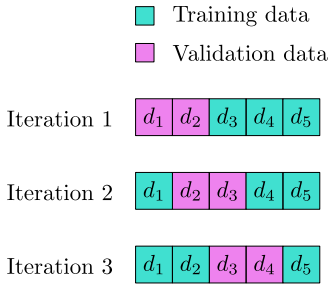

FIGURE 11. Training and validation data for the 3-fold cross-validation process.

One can notice in Table 2 that the mean accuracy of the classifier increases by combining the tuning of τ_u with the baseline model, as expected. The classifier with the baseline setup and τ_u tuning achieved $(93.1 \pm 0.7)\%$, representing an increase of 3% in the mean accuracy compared to the baseline performance. It is worth mentioning that this combination has the disadvantage of increasing the training time compared to the baseline setup due to cross-validation.

TABLE 3. Best threshold values τ_u for each subject obtained by the multi-holdout (1st row) and 3-fold cross-validation (2nd row) methods.

	Subject									
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
$\tau_{u,mh}$	11	13	15	15	16	18	13	10	13	19
$\tau_{u,cross}$	10	13	16	15	16	10	17	10	13	19

The second validation technique is the multi-holdout method, where we split the test dataset into two new datasets for each user. We used one dataset in the validation stage and the other for testing, i.e., a new test dataset. It is worth highlighting that both datasets have fixed data examples. In other words, as soon as the test dataset was split into two new ones, both remained unaltered throughout all the experiments. Then, we repeat the training and validation stages (holdout) 10 times for each hyperparameter value.

The experiment consists in splitting the test dataset $\mathcal{D}_{\text{test}}$ into $\mathcal{D}'_{\text{test}}$ and the validation set $\mathcal{D}_{\text{val}}$. Since we want the best test error estimate, we choose data from the test set to

make validation. The validation dataset has 6 examples of each class in $\mathcal{D}_{\text{test}}$, adding up to 30 data examples for each person. In other words, $\mathcal{D}_{\text{val}}$ is 20% of the original test set $\mathcal{D}_{\text{test}}$. It is worth mentioning that the examples we choose to compose the validation set are fixed, i.e., the same six data examples from each class in $\mathcal{D}_{\text{test}}$ for each person were separated to create $\mathcal{D}_{\text{val}}$, which makes the validation dataset $\mathcal{D}_{\text{val}}$ to be the same in all runs. Hence, we performed 10 runs of the training in $\mathcal{D}_{\text{train}}$ plus validation in $\mathcal{D}_{\text{val}}$, selected the model with the lowest validation error, and tested in the new testing dataset without the validation examples $\mathcal{D}'_{\text{test}}$. In this experiment, the topology of the FNN was changed to $6\times8\times6$. Finally, we computed the results by averaging the outcome of 100 independent trials.

TABLE 4. Mean accuracy and standard deviation results for different classification models; comparing the (baseline + $m700$ + 3fCV + *poll*) tested in $\mathcal{D}_{\text{test}}$; the (baseline + $m700$ + 3fCV + *poll*) tested in $\mathcal{D}'_{\text{test}}$; and the (baseline + $m700$ + *Mholdout* + *poll*) tested in $\mathcal{D}'_{\text{test}}$, where *Mholdout* indicates the multi-holdout validation method. All models used a $6 \times 8 \times 6$ neural network.

Model	Accuracy (%) (Mean)±(1 std)
baseline + $m700$ + 3fCV + *poll* + $\mathcal{D}_{\text{test}}$	97.0 ± 0.4
baseline + $m700$ + 3fCV + *poll* + $\mathcal{D}'_{\text{test}}$	96.9 ± 0.5
baseline + m700 + Mholdout + poll + $\mathcal{D}'_{\text{test}}$	$\mathbf{97.2 \pm 0.3}$

Table 4 shows the performance of the models that combine all the proposed modifications in combination with the new FNN architecture. We compared some overcomplete neural network architectures by varying the number of neurons in the hidden layer from 6 to 9. This search yielded the $6 \times 8 \times 6$ network as the best topology. Additionally, in order to compare the results, we designed two other models. The first model is the best in Table 2, which consists of combining all the proposed modifications but using a $6 \times 8 \times 6$ neural network. In the second model, we used $\mathcal{D}'_{\text{test}}$ instead of $\mathcal{D}_{\text{test}}$ for the testing. One may notice that both models achieved almost the same performance, but the model tested in $\mathcal{D}_{\text{test}}$ reached slightly better mean accuracy and standard deviation. However, the model using the multi-holdout validation method attained the best performance. It achieved a mean accuracy of 97.2% and a standard deviation of 0.3%. Although performing the best, this technique has the drawback of decreasing the test dataset, which can often be unfeasible since we have limited data. In a real-world application, such as the prosthetic arm problem, one may argue this represents the calibration stage, where we require the user to perform some gestures to fine-tune the hand response.

VI. DISCUSSION

The experiments presented in Section V consisted of modifying the baseline model by incremental changes in its structure, studying and comparing the impact of each modification in the classification system. The changes proposed were (i) training a unique τ_u for each subject to enhance the muscle activity detection; (ii) increasing the length of the time window during the testing stage; (iii) a new post-processing voting system (poll); (iv) a validation methodology; and (v) an overcomplete neural network topology.

In Table 2, we presented the results of our experiments. In those experiments, we combined some of the proposed modifications and tested them in all the 1500 testing examples. Hence, we can fairly compare the performance of each configuration with the baseline. One can notice that all the proposed models performed better than the baseline classifier, meaning that each of the proposed modifications was effective both when applied alone or when combined with other modifications. It is worth mentioning that the model combining all the proposed modifications achieved the highest mean accuracy and smallest standard deviation. Such a model reached results of $(96.5 \pm 0.4)\%$, outperforming the baseline model by a mean accuracy of 6.4%.

TABLE 5. Mean accuracy and standard deviation results per subject for the best model (baseline + $m700$ + *Mholdout* + *poll* + $6 \times 8 \times 6$ ANN).

Subject	Accuracy (%) (Mean)±(1 std)
#1	100 ± 0.0
#2	98.2 ± 0.4
#3	94.4 ± 1.8
#4	100 ± 0.0
#5	95.0 ± 1.5
#6	91.6 ± 1.6
#7	93.8 ± 1.1
#8	99.9 ± 0.2
#9	99.1 ± 0.2
#10	100 ± 0.0

In the last experiment, we evaluated the impact of the proposed multi-holdout validation method and the use of an overcomplete ANN with 8 neurons in the hidden layer. The only difference between the setup in the last row of Table 2 and the one in the first row of Table 4 is in the ANN. While the former uses 6 neurons in the hidden layer, the latter utilizes 8, improving the mean accuracy in 0.5%. It is worth mentioning that we also tried other values for the number of neurons in the hidden layer, but values greater than 8 yielded no significant gain in accuracy. Comparing the results presented in Table 4, one can notice that the setup with the multi-holdout validation method was superior, achieving mean accuracy and standard deviation of $(97.2 \pm 0.3)\%$, outperforming the baseline model by 7.1% in the mean accuracy. We can conclude that validating the model using data examples more similar to the testing ones contributes to the fine-tuning of τ_u. The multi-holdout provided a better solution to train such a hyperparameter considering our current datasets. By using this method, we also were able to achieve 100% of mean accuracy for some subjects, as depicted in Table 5. In Table 5, we show the mean accuracy and standard deviation for each subject, using the model with multi-holdout validation (baseline + $m700$ +*Mholdout*+ *poll*).

In Table 6, we show the mean and standard deviation of the inference time, the time the system takes to process and

TABLE 6. Mean inference time in milliseconds (ms) considering the testing time windows from Subject #1, where baseline represents the baseline model, $m700$ indicates that the window length was $m_{\text{test}} = 700$, and ANN $6 \times 8 \times 6$ refers to the model using an overcomplete $6 \times 8 \times 6$ ANN.

Model	Processing time (ms) (Mean)±(1 std)
baseline [23]	88.57 ± 38.58
baseline + $m700$	89.03 ± 38.07
baseline + ANN $6 \times 8 \times 6$	89.00 ± 37.93
baseline + $m700$ + ANN $6 \times 8 \times 6$	88.50 ± 37.84

TABLE 7. Mean post-processing time in milliseconds (ms) considering the testing data from Subject #1, where baseline represents the standard post-processing used in the baseline model, whereas *poll* is the proposed post-processing method.

Method	Processing time (ms) (Mean)±(1 std)
baseline [23]	0.30 ± 0.11
poll	0.30 ± 0.15

TABLE 8. Prediction accuracy reported in works that used data coming from the Myo armband.

Studies	Persons	Gestures	Accuracy(%)
Motoche [25]*	10	6	90.7
Yang [50]*	1	6	96.7
Zhang [51]	10	6	96.5
Dolopikos [52]	15	4	92.0
Chen [53]*	9	5	89.0
Javaid [54]	10	4	83.9
Our best model*	10	6	**97.2**

classify one time window (i.e., post-processing time is not included) during the test stage. For this purpose, we have arbitrarily chosen Subject #1, and we averaged the inference time over his 150 testing examples for several models. The inference time remained almost the same in every case. The inference time was (88.50 ± 37.84) ms when both modifications were combined. It is worth highlighting that for every window, the inference time of the proposed models was less than 300 ms, thus fulfilling the requirement of real-time processing [23], [32]. Additionally, [23] reported that the inference time of the baseline model in their hardware setup was only 11 ms, about 8 times lower than the mean processing time we obtained (approximately 88 ms), which indicates that there is room for processing time optimization in our codes. As for the post-processing techniques, both the baseline post-processing and the proposed poll achieved similar processing times, as reported in Table 7. All the results were generated using an Intel® Xeon® 2.20 GHz processor and 13 GB of RAM (no GPU was used).

Finally, in Table 8, we summarize the results that have been reported in recent works using sEMG data coming from the Myo armband which also classified some basic gestures, where (*) refers to works classifying the *fist*, *wave in*, *wave out*, *open*, and *pinch* gestures (i.e., the same gestures considered in this work). In addition to the reported accuracy, we also included the number of subjects and number of gestures considered in each work. One may notice that the prediction accuracy reported in our work (our best model) was the best one among those works.

VII. CONCLUSION

This article proposed an HGC system based on ANN that uses DTW-based features and enhanced muscle activity detection techniques to classify six gestures from ten healthy volunteers. The features were extracted from sEMG signals acquired via the Myo armband, which is a device that is widely used in the HGC field making it easier to compare results and techniques. Our pre-processing and feature extraction modules are a good approach to tackle the HGC problem since the combination of both methods results in a feature vector robust to the sensor's position and drift when compared to time or frequency-domain features, which most of the time require the sensors to be carefully placed over specific points (muscles). Validation was incorporated into the segmentation process in order to detect muscle contraction more accurately. Compared to the baseline model, our proposed system uses slightly larger testing windows so the testing examples are more similar to those in the training stage, uses an overcomplete ANN so the neural network can capture some complex patterns providing a richer representation of the data, and uses the poll method to enhance the post-processing. In summary, our paper proposes modifications in 4 out of the 5 modules of the baseline model. Detailed experiments with real data were presented comparing different classification models and the impact of each modification. The results showed that each of the proposed modifications in the baseline model was effective, increasing the mean accuracy while maintaining low inference time. Indeed, the model with all the proposed techniques achieved the best mean accuracy result (97.2%), surpassing the baseline model significantly.

Regarding future works, we will study the viability of using other feature sets as well as other methods to calculate the DTW. For instance, there is a promising implementation of the DTW distance which, in our preliminary results, led to the same accuracy while reducing the processing time by about 10 times, in comparison to the implementation we used in this work.

REFERENCES

[1] M. Gonzalez-Fernandez. *Amputation: Recovery and Rehabilitation*. Accessed: Jun. 2, 2022. [Online]. Available: https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/amputation/amputation-recovery-and-rehabilitation

[2] T. W. Wright, A. D. Hagen, and M. B. Wood, "Prosthetic usage in major upper extremity amputations," *J. Hand Surg.*, vol. 20, no. 4, pp. 619–622, Jul. 1995.

[3] American Academy of Orthopaedic Surgeons, *Atlas of Limb Prosthetics: Surgical, Prosthetic, and Rehabilitation Principles* (Medical, Physical Medicine & Rehabilitation), 2nd ed., Maryland Heights, MI, USA: Mosby Year Book, 1992.

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

[5] M. E. Huang, C. E. Levy, and J. B. Webster, "Acquired limb deficiencies. 3. Prosthetic components, prescriptions, and indications," *Arch. Phys. Med. Rehabil.*, vol. 82, no. 3, pp. S17–S24, Mar. 2001.
[6] 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 Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 4, pp. 797–809, Jul. 2014.
[7] E. A. Clancy, E. L. Morin, and R. Merletti, "Sampling, noise-reduction and amplitude estimation issues in surface electromyography," *J. Electromyogr. Kinesiol.*, vol. 12, no. 1, pp. 1–16, Feb. 2002.
[8] 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 Neurosci.*, vol. 15, pp. 1–20, Apr. 2021. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnins.2021.621885
[9] D. Farina and O. Aszmann, "Bionic limbs: Clinical reality and academic promises," *Sci. Transl. Med.*, vol. 6, no. 257, p. 257, Oct. 2014.
[10] E. J. Scheme, K. B. Englehart, and B. S. Hudgins, "Selective classification for improved robustness of myoelectric control under nonideal conditions," *IEEE Trans. Biomed. Eng.*, vol. 58, no. 6, pp. 1698–1705, Jun. 2011.
[11] C. Nissler, N. Mouriki, and C. Castellini, "Optical myography: Detecting finger movements by looking at the forearm," *Frontiers Neurorobot.*, vol. 10, pp. 1–10, Apr. 2016. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fnbot.2016.00003
[12] 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 Trans. Neural Syst. Rehabil. Eng.*, vol. 24, no. 7, pp. 744–753, Jul. 2016.
[13] 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 Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 3, pp. 501–510, May 2014.
[14] 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, p. 2467, Apr. 2020. [Online]. Available: https://www.mdpi.com/1424-8220/20/9/2467
[15] N. M. Hye, U. Hany, S. Chakravarty, L. Akter, and I. Ahmed, "Artificial intelligence for sEMG-based muscular movement recognition for hand prosthesis," *IEEE Access*, vol. 11, pp. 38850–38863, 2023.
[16] A. Sadiq, S. G. Khawaja, M. U. Akram, N. S. Alghamdi, A. Khan, and A. Shaukat, "Machine learning and signal processing based analysis of sEMG signals for daily action classification," *IEEE Access*, vol. 10, pp. 40506–40516, 2022.
[17] A. Tigrini, F. Verdini, M. Scattolini, F. Barbarossa, L. Burattini, M. Morettini, S. Fioretti, and A. Mengarelli, "Handwritten digits recognition from sEMG: Electrodes location and feature selection," *IEEE Access*, vol. 11, pp. 58006–58015, 2023.
[18] D. Cho, Y.-W. Tai, and I. S. Kweon, "Deep convolutional neural network for natural image matting using initial alpha mattes," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1054–1067, Mar. 2019.
[19] X. Tang, "Hybrid hidden Markov model and artificial neural network for automatic speech recognition," in *Proc. Pacific-Asia Conf. Circuits, Commun. Syst.*, May 2009, pp. 682–685.
[20] F. Z. Xing, E. Cambria, and R. E. Welsch, "Natural language based financial forecasting: A survey," *Artif. Intell. Rev.*, vol. 50, no. 1, pp. 49–73, Jun. 2018.
[21] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, Nov. 2018, Art. no. e00938.
[22] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017.
[23] 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 *Proc. 26th Eur. Signal Process. Conf. (EUSIPCO)*, Sep. 2018, pp. 1492–1496.
[24] G. J. Rani, M. F. Hashmi, and A. Gupta, "Surface electromyography and artificial intelligence for human activity recognition—A systematic review on methods, emerging trends applications, challenges, and future implementation," *IEEE Access*, vol. 11, pp. 105140–105169, 2023.
[25] C. Motoche and M. E. Benalcázar, "Real-time hand gesture recognition based on electromyographic signals and artificial neural networks," in *Proc. 27th Int. Conf. Artif. Neural Netw. (ICANN)*, Rhodes, Greece. Cham, Switzerland: Springer, Oct. 2018, pp. 352–361.
[26] F. Al Omari, J. Hui, C. Mei, and G. Liu, "Pattern recognition of eight hand motions using feature extraction of forearm EMG signal," *Proc. Nat. Acad. Sci., India A, Phys. Sci.*, vol. 84, no. 3, pp. 473–480, Sep. 2014.
[27] S. M. Mane, R. A. Kambli, F. S. Kazi, and N. M. Singh, "Hand motion recognition from single channel surface EMG using wavelet & artificial neural network," *Proc. Comput. Sci.*, vol. 49, pp. 58–65, Jan. 2015.
[28] X. Liu, J. Sacks, M. Zhang, A. G. Richardson, T. H. Lucas, and J. Van der Spiegel, "The virtual trackpad: An electromyography-based, wireless, real-time, low-power, embedded hand-gesture-recognition system using an event-driven artificial neural network," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 64, no. 11, pp. 1257–1261, Nov. 2017.
[29] E. A. Chung and M. E. Benalcázar, "Real-time hand gesture recognition model using deep learning techniques and EMG signals," in *Proc. 27th Eur. Signal Process. Conf. (EUSIPCO)*, Sep. 2019, pp. 1–5.
[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (Adaptive Computation and Machine Learning Series). Cambridge, U.K.: MIT Press, 2016.
[31] X. Chen and Z. J. Wang, "Pattern recognition of number gestures based on a wireless surface EMG system," *Biomed. Signal Process. Control*, vol. 8, no. 2, pp. 184–192, Mar. 2013.
[32] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE Trans. Biomed. Eng.*, vol. 40, no. 1, pp. 82–94, Jan. 1993.
[33] 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," *Sci. Data*, vol. 1, no. 1, pp. 1–13, Dec. 2014.
[34] 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 Comput. Appl.*, vol. 35, no. 22, pp. 16101–16111, Aug. 2023.
[35] S. Thusneyapan and G. I. Zahalak, "A practical electrode-array myoprocessor for surface electromyography," *IEEE Trans. Biomed. Eng.*, vol. 36, no. 2, pp. 295–299, Feb. 1989.
[36] V. T. Inman, H. J. Ralston, J. B. De C. M. Saunders, M. B. B. Feinstein, and E. W. Wright, "Relation of human electromyogram to muscular tension," *Electroencephalogr. Clin. Neurophysiol.*, vol. 4, no. 2, pp. 187–194, May 1952. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0013469452900084
[37] J. G. Kreifeldt, "Signal versus noise characteristics of filtered EMG used as a control source," *IEEE Trans. Biomed. Eng.*, vol. BME-18, no. 1, pp. 16–22, Jan. 1971.
[38] Y. St-Amant, D. Rancourt, and E. A. Clancy, "Effect of smoothing window length on RMS EMG amplitude estimates," in *Proc. IEEE 22nd Annu. Northeast Bioeng. Conf.*, Mar. 1996, pp. 93–94.
[39] P. S. R. Diniz, E. A. B. da Silva, and S. L. Netto, *Digital Signal Processing: System Analysis and Design*, 2nd ed., Cambridge, U.K.: Cambridge Univ. Press, 2010.
[40] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed., Upper Saddle River, NJ, USA: Prentice-Hall, 2009.
[41] S. K. Mitra, *Digital Signal Processing: A Computer Based Approach*, 1st ed., New York, NY, USA: McGraw-Hill, 1997.
[42] M. H. Hayes, *Statistical Digital Signal Processing and Modeling*, 1st ed., Hoboken, NJ, USA: Wiley, 1996.
[43] S. Haykin, *Communication Systems*, 5th ed., Hoboken, NJ, USA: Wiley, 2009.
[44] M. Müller, *Information Retrieval for Music and Motion*. Berlin, Germany: Springer-Verlag, 2007.
[45] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed., Upper Saddle River, NJ, USA: Prentice-Hall, 2004.
[46] Y. S. Abu-Mostafa, M. Magdon-Ismail, and H.-T. Lin, *Learning From Data: A Short Course*. CA, USA: AMLBook, 2012.
[47] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*, vol. 4. Cham, Switzerland: Springer, 2006.
[48] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *Proc. Int. Joint Conf. Artif. Intell.*, vol. 14. Montreal, QC, Canada, 1995, pp. 1137–1145.
[49] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd ed., Hoboken, NJ, USA: Wiley, 2000.

[50] K. Yang and Z. Zhang, "Real-time pattern recognition for hand gesture based on ANN and surface EMG," in *Proc. IEEE 8th Joint Int. Inf. Technol. Artif. Intell. Conf. (ITAIC)*, May 2019, pp. 799–802.
[51] Z. Zhang, Q. Shen, and Y. Wang, "Electromyographic hand gesture recognition using convolutional neural network with multi-attention," *Biomed. Signal Process. Control*, vol. 91, May 2024, Art. no. 105935. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S174680942301368X
[52] C. Dolopikos, M. Pritchard, J. J. Bird, and D. R. Faria, "Electromyography signal-based gesture recognition for human–machine interaction in real-time through model calibration," in *Advances in Information and Communication*, K. Arai, Ed., Cham, Switzerland: Springer, 2021, pp. 898–914.
[53] W. Chen and Z. Zhang, "Hand gesture recognition using sEMG signals based on support vector machine," in *Proc. IEEE 8th Joint Int. Inf. Technol. Artif. Intell. Conf. (ITAIC)*, May 2019, pp. 230–234.
[54] H. A. Javaid, M. I. Tiwana, A. Alsanad, J. Iqbal, M. T. Riaz, S. Ahmad, and F. A. Almisned, "Classification of hand movements using MYO armband on an embedded platform," *Electronics*, vol. 10, no. 11, p. 1322, May 2021. [Online]. Available: https://www.mdpi.com/2079-9292/10/11/1322

GABRIEL S. CHAVES (Member, IEEE) was born in Rio de Janeiro, Brazil, in 1994. He received the Telecommunications Engineering degree from Universidade Federal Fluminense (UFF), in 2016, and the M.Sc. degree in electrical engineering from COPPE/UFRJ, in 2019, where he is currently pursuing the Ph.D. degree in electrical engineering. His research interests include digital signal processing, machine learning, artificial intelligence, computer vision, biomedical signals, and biomedical engineering. He is a member of Brazilian Telecommunications Society (SBrT). He was awarded as the Best Undergraduate Student of the Year from Universidade Federal Fluminense (UFF), in 2016. He received the Best Paper Award at Brazilian Conference in Computational Intelligence (CBIC 2023).

ANDERSON S. VIEIRA was born in Fortaleza, Brazil, in 1993. He received the bachelor's degree (magna cum laude) from Universidade Federal do Ceará (UFC), in 2017, and the M.Sc. degree in electrical engineering from COPPE/UFRJ, in 2021. His research interests include digital signal processing, machine learning, and optimization.

MARKUS V. S. LIMA (Member, IEEE) received the B.Sc. degree in electronics engineering from the Federal University of Rio de Janeiro (UFRJ), in 2008, and the M.Sc. and D.Sc. degrees in electrical engineering from COPPE/UFRJ, in 2009 and 2013, respectively. Currently, he is an Associate Professor with the Program of Electrical Engineering (PEE/COPPE) and the Department of Electrical Engineering (DEE/Poli), UFRJ. He has co-authored the books *Online Learning and Adaptive Filters* and *Block Transceivers: OFDM and Beyond*. His research interests include machine learning theory and applications, adaptive filters, and digital signal processing. He is a member of the editorial board of *Digital Signal Processing* (Elsevier) and IEEE Open Journal of Signal Processing. He received the Best Paper Award at Brazilian Conference in Computational Intelligence (CBIC 2023). He has integrated the technical program committee (TPC) of several conferences and the board of the IEEE Section in Rio de Janeiro, from 2016 to 2017. Currently, he is the Chair of the IEEE Signal Processing Chapter in Rio de Janeiro.