Chapter 12
Application Study of Electroencephalographic Signals in the Upper Limb Prosthesis Field

Alexandre Ferreira Kleppa, Anderson Luis Szejka, and Leandro dos Santos Coelho

12.1 Introduction

Loss or being born without limbs can significantly affect the level of autonomy and the capability of performing daily living, working and social activities (Cordella et al. 2016). There are many reasons why a person may lose a limb. Congenital circumstances, diseases, industrial and car accidents are among the other causes of lost limbs. In the United States, the most common cause of amputation is diabetes (Geiss et al. 2019), and other researches show that this situation is not limited to the United States of America (Santos et al. 2006; Laclé and Valero-Juan 2012; Hoffstad et al. 2015). Based on this context, current research is seeking new methods and technologies to improve prosthetic limbs manufacturing, usability, flexibility, and versatility to propitiate the most suitable welfare for the user (Canciglieri et al. 2019; Kumar et al. 2019; Kashef et al. 2020).

Prosthetic hands are a type of artificial limb that has received research attention in recent years (Kashef et al. 2020; Badawy and Alfred 2020; Graham et al. 2021; Vaskov and Chestek 2021). The use of prosthetic hand to replace the loss is desirable,

A. F. Kleppa
Control and Automation Engineering Department, Pontifical Catholic University of Parana (PUCPR), Curitiba, Parana, Brazil
e-mail: alexandre.kleppa@pucpr.edu.br

A. L. Szejka (✉) · L. dos Santos Coelho
Industrial and Systems Engineering Graduate Program, Pontifical Catholic University of Parana, Curitiba, Parana, Brazil
e-mail: anderson.szejka@pucpr.br

L. dos Santos Coelho
e-mail: leandro.coelho@pucpr.br

L. dos Santos Coelho
Department of Electrical Engineering (PPGEE), Federal University of Parana (UFPR), Polytechnic Center, Curitiba, Brazil

O. Canciglieri Junior and M. D. Trajanovic (eds.), *Personalized Orthopedics*,
https://doi.org/10.1007/978-3-030-98279-9_12

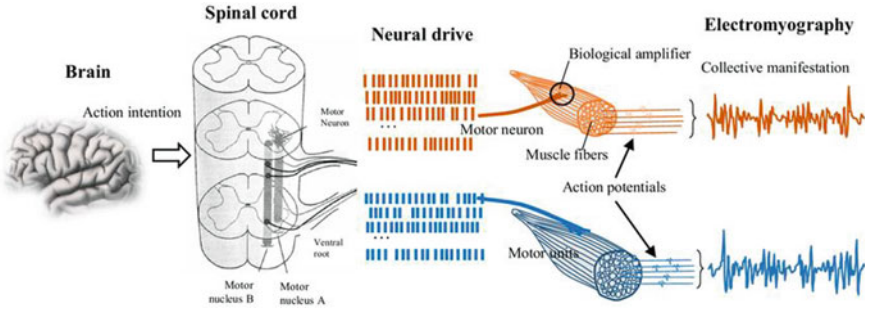

Fig. 12.1 The generation of EMG signals (Yang et al. 2019)

with modern prosthetic hands, which it has evolved from the mechanical nails of yesteryears to sophisticated devices that are electrically powered and offer large numbers of Degrees of Freedom (DOF) (Kumar et al. 2019). The control of the first prosthetic hands was merely mechanical, where the user had to wear a mechanism and when the arm would move in a certain way (Badawy and Alfred 2020). As prosthetic hand presents a large number of DOF and open and close with force and position control, new prosthetic hand control methods are required (Vaskov and Chestek 2021).

The modern hand's owners can perform the control of the hand using electrodes that would give the hand's controller input (Badawy and Alfred 2020). This type of prosthetic control gets a bit more natural, once that the electrode receives the electric signal from the user's muscle and gives it to the system as an output to open or close the hand (Eisenberg et al. 2017). However, the control of the hand using electrodes is complex and require technologies like electromyography (EMG) and modern algorithms to treat the signal and convert it into an output to control the hand movement (Eisenberg et al. 2017; Badawy and Alfred 2020; Graham et al. 2021; Vaskov and Chestek 2021). EMG technology reads the impulse sent by the brain through the spinal cord to the muscles as can be seen in Fig. 12.1.

According to Fig. 12.1, the brain also sent an electric impulse that is generated in the frontal-lobe motor areas that were carried through the spinal cord and sent to the muscle (in this case could be the muscle in the residual limb). The difference from the limb-powered one is that, in this case, the signal will be read by the electrodes placed on the patient's residual limb, and this signal will be used as the information to be processed then moving the prosthesis.

Even though the commercial upper limb electric-powered prosthesis available in the market use in its big majority EMG, the residual limb has muscular limitations, especially the ones after an above-elbow amputation what makes the EMG not enough for the control of prosthesis with multiple degrees of freedom. To solve this problem, there is the proposal of using signal fusion, which is using EMG combining with Electroencephalography (EEG) for having more trustful data (Li et al. 2017). EEG is a method of brain exploration that measures electrical activity in the brain through electrodes placed on the scalp often shown as a trace called an electroencephalogram

(Srinivasan and Nunez 2012). Recently, researchers showed a success rate of 68% using EEG signals to trigger the tasks in a robotic hand system (Saint-Elme et al. 2017; Eisenberg et al. 2017; Badawy and Alfred 2020; Graham et al. 2021; Vaskov and Chestek 2021), which also shows the promising future of the EEG research field.

According to this context, the chapter explores an approach that integrates Principal Component Analysis (PCA) and Recurrent Neural Network (RNN) techniques to evaluate the performance in control a prosthetic hand based on the signals generated from the user's intention of movement recorded from electroencephalographic (EEG). Additionally, the Long-Short Term Memory (LSTM) is a model of RNN of the deep learning field, which will evaluate the performance in face of different Machine Learning methods to control the movement of a prosthetic hand.

The remainder of the paper is structured as follows: Sections 12.2–12.4 explores the related works about the Brain-Computer Interface (BCI), PCA and RNN to support the problem definition. Section 12.5 is dedicated to conceptualizing the integration of PCA with LSTM-RNN to control a prosthetic hand. Section 12.6 presents the application of the approach in an experimental case. Section 12.7 discusses the results, main advantages, and limitations of the research.

12.2 Brain-Computer Interface (BCI)

The BCI is the technique that uses electric signals spotted from the scalp, cortical surface, or brain subcortical areas to activate external devices (Vaskov and Chestek 2021). The electric signals can be captured in an invasive manner or a non-invasive one. The first one is done by the register of a small or big number of neurons, the non-invasive manner is to capture and monitoring the signals by electroencephalography (EEG) (Nuwer and Coutin-Churchman 2014; Srinivasan and Nunez 2012; Machado et al. 2009).

A BCI has, normally, the following components: signal acquisition, preprocessing, feature extraction, classification (or detection), and the application of the interface (Graimann et al. 2010). The explanation can be found in the sequence:

- *Signal acquisition*: This part is responsible for recording the electrophysiological signals that are going to be the input to the BCI.
- *Preprocessing*: This is the task that must enhance the signal-to-noise ratio. To accomplish this task advanced signal processing methods can be useful.
- *Feature extraction*: The objective in this stage is to find an adequate representation of the obtained signals in the previous steps that simplify the classification or detections of the patterns that are going to be studied.
- *Classification*: The goal in this phase is to use the signal features obtained by the previous one to assign the recorded samples into a category of brain patterns.
- *Application of the interface*: As the name may suggest, here is where the BCI system is applied. For advanced applications of the BCI system, it can be found

the output as the controlling of spelling systems or some external devices as prosthetic or multimedia applications.

There is a big interest in this area, once that the implementation of the technique can help to compensate for a motor control loss of patients with a degree more severe of limitation. BCI can be a good indication for patients that suffer from lateral sclerosis amyotrophic, spinal cord injury, stroke, and cerebral palsy, for example, once that BCI captures and modifies the signals that come from brain activity (intension of a movement, for instance) into action, what allows the patient to communicate with the external world (Machado et al. 2009).

When many neurons are activated in the brain, an electric field is formed, EEG is the method used to record and monitor it. Those recorded signals can reveal a data set on cognitive processing, for instance, the intention of a movement (Bäckström and Tidare 2016). To obtain those signals, a technique called electroencephalogram can be done. It is a recording of the brain's electrical potentials done by a set of electrodes placed on the patient scalp (Nunez and Srinivasan 2006).

The electric signals from the brain can also be obtained by intracranial electrodes, implanting them on the animals or epileptic patients for instance, however, as said by Nunez and Srinivasan (2006), although in the second option more detailed and local information can be recorded, it fails to record the general picture of the brain activity. Therefore, in practice, intracranial recordings can just give a different data set of information than the one that can be obtained by the scalp electrodes, not a better one.

According to Bäckström and Tidare (2016), the technique of using EEG signals to control devices through a brain-computer interface is being used for more than a decade now. The appliance the technique resulted, for example, in movement computer cursor, the control of a mobile robot and a virtual keyboard, and some more examples.

12.3 Principal Component Analysis (PCA)

As previously exposed, feature extraction is one of BCI's steps. EEG data can be a large dataset difficult to analyze. To be able to study the EEG data, it can be approached and represented in another way.

Dimensionally reduction is one processing step used to transform features into lower dimension space. One of the most famous unsupervised techniques is the PCA. It is a technique that has the goal of finding the space that better represents the direction of the maximum variance of the data (Tharwat 2016). The PCA has several purposes: finding relationships between observations, extracting the main information from the data, outlier detection and removal, and selecting only the important information of the data reducing its dimensions. The principal components space is composed of orthogonal principal components that are calculated using the

covariance matrix or Singular Value Decomposition (SVD) (Tharwat 2016). It can perform dimensionally reduction without any loss of data (Iftikhar et al. 2018).

To use PCA in this study, a background is required, and some concepts must be consolidated. The following sections present a review of the theory that precedes the study of the PCA.

12.3.1 Standard Deviation (s) and Variance (S2)

It is not possible to analyze the features of an entire population, for this reason, studies are normally done using a sample (Tharwat 2016). When a sample of a population is taken for a study, the standard deviation is used to analyze how to spread the individual points of this sample are from the mean. It is a basic statistics technique but will be used in the following steps during the development of the PCA technique. Standard Deviation is calculated by Eq. (12.1) given by

$$s = \sqrt{\frac{\sum_{i=1}^{n} X_i - \overline{X}}{N-1}} \tag{12.1}$$

where X_i is the value in the data that refers to the point i, X is the mean of the data and N is the number of data points in the population. It shows how distant an observation is from the distribution center. Therefore, if the standard deviation has a great value, it tells that the data is very spread out, while a low standard deviation shows that the data has a small variability. The variance is also a way of verifying the vary between the points of the sample and the mean. Its formula is given by the standard deviation squared (s2). It measures the deviation of the variable from its mean value, which is the standard deviation squared with

$$s^2 = \frac{\sum_{i=1}^{n} X_i - \overline{X}}{N-1} \tag{12.2}$$

where X_i is the value in the data that refers to the point i, X is the mean of the data and N is the number of the data points in the population.

12.3.2 Covariance (Cov) and Covariance Matrix (Σ)

The standard deviation and the variance are technics that can analyze the characteristics of the attributes of the population's sample only in one dimension, which means that it cannot check the relation between two or more attributes of the sample, however, many times can be interesting to check the influence that they have on each other. For this task, a good solution is calculating the covariance between them. The

covariance is measured always in two dimensions, and it is given by

$$cov(X, Y) = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{N - 1} \tag{12.3}$$

where X_i is the value in the data that refers to the point i in the attribute X, X is the mean of the data in the attribute X, N is the number of the data points in the population, Y_i is the value in the data that refers to the point i in the attribute Y, Y is the mean of the data in the attribute Y and $co(X,Y)$ is the representation of the covariance between X and Y. If the result is a negative covariance, it means that when one value increases the other decreases. If the covariance is positive, it means that the values of both attributes increase or decrease in a direct proportion. In the case that the result is equal to zero, the interpretation is that the attributes have no relation between them.

Covariance shows if there is a relation between two dimensions, if there are more than two dimensions in one data set, the covariation matrix can be used. The covariation matrix is given by

$$C^{xnx} = (c_{i,j}, c_{i,j} = cov(Dim_i, Dim_j)) \tag{12.4}$$

From this equation, considering a hypothetical dataset with the dimensions (x, y, z), the possibilities of covariance analyses would be $co(x,y)$, $cov(x,z)$, $cov(y,z)$, and the covariances between each dimension and itself. Therefore, in this hypothetical case, the covariance matrix is given by

$$\Sigma = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix} \tag{12.5}$$

Calculating the eigenvalues and eigenvectors is the way to solve the covariance matrix (Σ) as shown in Eq. (12.6). Eigenvalues are scalar values and eigenvectors are non-zero vectors, they represent the principal components explained in sequence.

$$V\Sigma = \lambda V \tag{12.6}$$

12.3.3 Calculating of Principal Component Analysis

The principal components (PCs) correspond to the direction of the maximum variance. They are orthonormal and uncorrelated. The PCA space corresponds to n PCs (Tharwat 2016). The first PC represents the direction of the maximum variance of the data (which is the eigenvector with the highest eigenvalue), the second one represents

the second largest variance in the data's direction (which is the eigenvector with the second-highest eigenvalue) and so on.

The eigenvectors are ordered considering the eigenvalue (highest to lowest). The eigenvector with the highest eigenvalue will be the principal component of e analysis, and from the order don before, eigenvectors with the lowest eigenvalues can be eliminated without concern for a loss on the data set. With the eigenvectors selected, a feature vector can be done, and a matrix is formed with all of them in the columns given by

$$FeatureVector = (eigen_1, eigen_2, eingen_3, \ldots, eigen_n) \tag{12.7}$$

For good feature extraction and the decision of how many principal components to use, a bar plot can be done, it helps to graphically analyze the PCs and decide which ones are the important ones, how many are needed to represent the biggest variance on the data.

In this case, in both examples the PCA1 and PCA2 would be used, the other ones do not contain much valuable information and can be discarded. After those previous steps, the new data set will be the product between the feature vector transposed and t data adjusted by the mean (transposed as well) given by

$$FinalData = (FeatureVector)^T (DataAdjusted)^T \tag{12.8}$$

These steps will give the output of the original data but in terms of the desired vectors. And now the data points are classified as a combination of the contributions from each of the lines.

12.4 Recurrent Neural Network

Many of the advanced artificial intelligence architectures today are inspired by Recurrent neural networks (RNNs). RNN are a special class of Artificial Neural Networks characterized by internal self-connections for feedback, unlike a traditional feedforward neural network (Bianchi et al. 2017). This feedback loop permits the RNN to model the effects of the earlier parts of the sequence on the later part of the sequence, which is a relevant aspect when it comes to modelling sequences (Hiriyannaiah et al. 2020).

RNN uses an internal state to perform a task (Reimers and Requena-Mesa 2020). Since the new internal state is calculated from the old internal state and the input, it can be understood as one part of the output of the neural network. A RNN presents multiple architectures such as Integer-only Recurrent Neural Network (iRNN); np-Recurrent Neural Network (np-RNN), Long Short Term Memor (LSTM) and Convolutional Neural Network (CNN).

LSTM is very used with short- and long-term dependencies in data. LSTM tries to solve the vanishing gradient problem by not imposing any bias towards recent

observations, but it keeps constant error flowing back through time (Reimers and Requena-Mesa 2020). LSTM has been employed in numerous sequence learning applications, especially in the field of classification. an LSTM cell is composed of 5 different nonlinear components, interacting with each other in a particular way. The internal state of a cell is modified by the LSTM only through linear interaction. To control the behavior of each gate, a set of parameters are trained with gradient descent, to solve a target task. Equation (12.9) presents the LSTM modelling (Bianchi et al. 2017) with

$$\begin{aligned}
&forgetgate : \sigma_f[t] = \sigma(W_f x[t] + R_f y[t-1] + b_f,\\
&candidatestate : \tilde{h}[t] = g_1(W_h x[t] + R_h y[t-1] + b_h,\\
&updatestate : \sigma_u[t] = \sigma(W_u x[t] + R_u y[t-1] + b_u,\\
&cellstate : h[t] = \sigma_u[t] \odot \tilde{h}[t] + \sigma_f[t] \odot h[t-1],\\
&outputgate : \sigma_o[t] = \sigma(W_o x[t] + R_o y[t-1] + b_o,\\
&output : y[t] = \sigma_o[t] \odot g_2(h[t])
\end{aligned} \tag{12.9}$$

where $x[t]$ is the input vector at time t. W_f, W_h, W_u, $and W_o$ are rectangular weight matrices, that are applied to the input of the LSTM cell. R_f, R_h, R_u, and R_o are square matrices that define the weights of the recurrent connections, while b_f, b_h, b_u, and b_o are bias vectors. The function $\sigma(\cdot)$ is a sigmoid 2, while $g1(\cdot)$ and $g2(\cdot)$ are pointwise non-linear activation functions, usually implemented as hyperbolic tangents that squash the values in $[-1, 1]$. Finally, $\odot$ is the entry wise multiplication between two vectors (Hadamard product).

12.5 PCA + LSTM-RNN Approach to Control a Prosthetic Hand Based on EEG

As presented in Sect. 12.3, PCA is a technique that has the goal of finding the space that better represents the direction of the maximum variance of the data. In this research, PCA is important since EEG signals are acquired in many channels, and some of them can be not relevant for studying and/or may increase the noise in the dataset. PCA reduces the dimension of the dataset using feature extraction, maintaining just the relevant information for a posterior analysis or classification.

After the PCA, it is necessary to classify the dataset to define the movement which the user intends to carry out. LSTM is an architecture of the neural of RNN that "remembers" the values in arbitrary intervals, and it is adequate to classify temporal series with non-known interval's durations (Lin et al. 2021) that is the case of EEG signals. In this way, this work is going to use an LSTM architecture for classifying the dataset.

Based on this context, a PCA + LSTM-RNN approach to control a prosthetic hand based on EEG was proposed in Fig. 12.2. The approach is divided into four

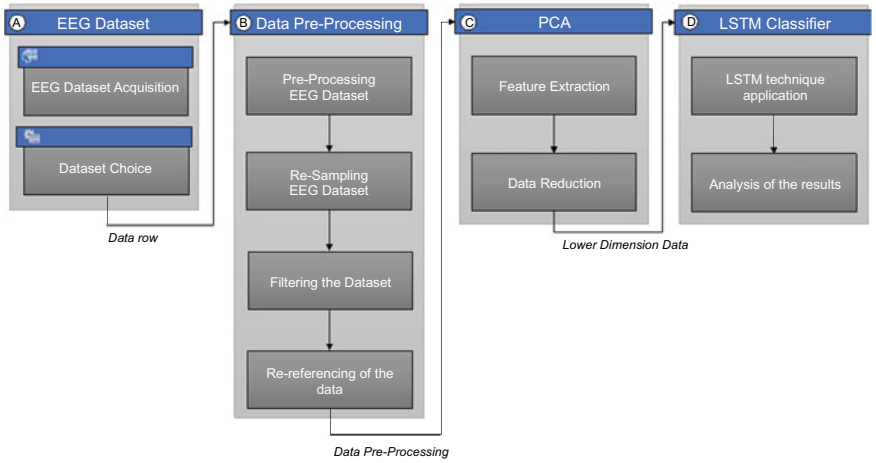

Fig. 12.2 PCA + LSTM-RNN approach to control a prosthetic hand

sections: EEG Dataset Acquisition (Detail "A"), Data Pre-Processing (Detail "B"), PCA (Detail "C"), LSTM Classifier (Detail "D").

In Fig. 12.2, the beginning of the process (Detail "A") can be seen where the data must be acquired through an already existent EEG database. The EEG database must go to the preprocessing part (Detail "B") where the sampling rate is adjusted, and the data is filtered and re-referenced. After this step, the PCA method (Detail "C") is applied, where the features will be extracted, and the data dimensions reduced. In the last part, an LSTM-RNN classifier (Detail "D") was applied and compared with other classifiers to verify the performance and accuracy to understand what kind of movement is necessary to carry out in a prosthetic hand.

12.5.1 Electroencephalographic Signals Database

In the last decades, many works, and research have been developed in the EEG and brain-computer interface field, as discussed in Sect. 12.2. For developing those studies, normally EEG signals were acquired from commercial devices, or other types of devices developed for this finality (Badawy and Alfred 2020; Vaskov and Chestek 2021).

Many of those datasets previously used for research or other applications are available online. On the internet and in related researches, EEG databases can be found, such as motor-imagery (Al-Saegh et al. 2021), emotion-recognition (Gannouni et al. 2021), error-related potentials (Keyl et al. 2019), visual-evoked potentials (Salelkar and Ray 2020), event-related potentials (Kropotov 2016), resting state (Corchs et al. 2019), EEG signals based in the subject's eye-blink (Sovierzoski et al. 2008) and so on.

The purpose of this work is to analyze the EEG signals and checking the possibility of them being the input of a robotic hand's control. Therefore, it is not necessary to acquire a new dataset of EEG and an online dataset is going to be selected. This option will let this study focus on the data processing and analysis, not carrying, in this first moment, for the data acquisition itself.

12.5.2 Data Preprocessing

The EEG recorded is called raw data. This data comes without any filtering or treatment. For the analysis that is going to be done in the next steps, it may be transformed into a format that is better to analyze and work with, this transformation step is called preprocessing.

The signals obtained from EEG are not only significant data, but the spatial information of the brain activity may also have got lost and they can be contaminated by muscle movement or blinking. Additionally, some noise in the data may obscure some weak EEG signals. Irrelevant activity is mixed with relevant activity and more possible problems can contaminate the data. To send a better dataset to the next steps some filtering techniques must be applied.

First, changing the sampling rate of the data is necessary to save the memory and the disk storage. Also, a good data sampling must be selected in this step for the processing time. The EEG recorded data comes in many different frequencies, therefore, many kinds of waves (alpha, beta, theta, and delta). The type of wave that better suits the project is going to be chosen, from that, to work only with the desired ones and a bandpass filter shall be applied.

In EEG a reference electrode must be chosen, the voltage of the other electrodes will be relative to the voltage of the reference one. If the reference electrode was not well chosen, it will contaminate the data, so a re-reference must be done in the preprocessing to send better data to the next steps. With these techniques, it is expected that the preprocessing part of the project will be good if an analysis shows that it not being enough, more techniques can be studied and applied.

12.5.3 PCA Application

After the application of the preprocessing method, the data is already improved for the final user. However, there is still some noise that can interpret this data as a difficult task. This noise must be removed. The feature meaningful features need to be extracted, between many techniques that could be used, such as Common Spatial Patterns, Wavelet Transform, Independent Components, and PCA. The PCA was chosen for two main reasons: (i) it is proven to be a relevant feature extraction method from many researches including (Lekshmi et al. 2014) (Subasi and Ismail

Gursoy 2010) and (Saint-Elme et al. 2017), and (ii) the interest of the author for this specific method.

PCA is a wide method used for pattern recognition and feature extraction (Lekshmi et al. 2014). It can be used in cases where exist a big dimension dataset with some variable redundancy can be reduced to a smaller number of principal components. In the case of this study, the redundant data can be eliminated using PCA. After applying the PCA method, it will be possible to analyze the data in a dimension reduced space, which will facilitate the classification of the variables in the next method.

12.5.4 Data Classification

After the signal is acquired, preprocessed and the features are extracted, the data must be classified before going to the application as an input.

Through the vast area of machine learning and deep learning, the chosen method for classifying the dataset in this project is one type of neural network called RNN. It is a type of neural network where the output of previous steps is fed as an input to the current step. In this model, the neural network allows the information to persist, a recurrent neural network can be thought of as a stack of multiple normal neural networks passing the information to one another (Silva et al. 2018; Modaresi et al. 2018; Lin et al. 2021). An illustration of an RNN can be seen in Fig. 12.3.

Where *A* is the hidden layers of the RNN, *X* the inputs, and *h* the outputs. LSTM is an architecture of the neural of RNN that "remembers" the values in arbitrary intervals, and it is adequate to classify temporal series with non-known interval's durations. To compare LSTM's accuracy with other methods, some Machines Learn methods were chosen, methods as Linear Discriminant Analysis (LDA), Random Forest, AdaBoost (Adaptive Boosting), and Support Vector Machines (SVM).

LDA is a classifier with a linear decision boundary, the classifier is generated by fitting class conditional densities to the data using Bayes's rule. Random Forest

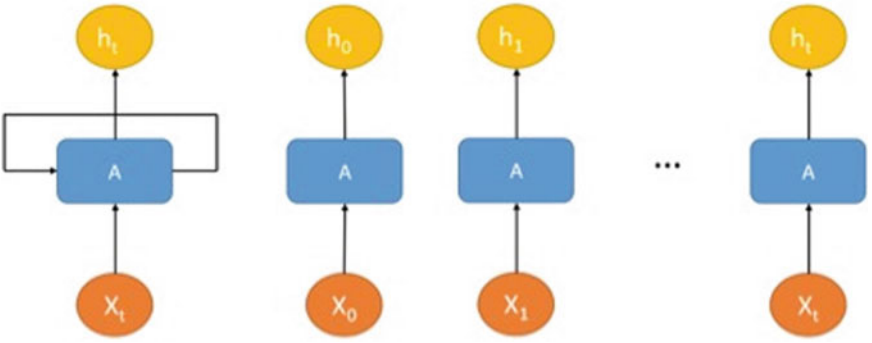

Fig. 12.3 Representation of an RNN

classifier is a machine learning method that fits several decision trees classifiers on various sub-samples of the dataset. Adaboost classifier is a method that works by fitting a classifier on the original dataset and, after that, fitting additional copies of the classifier on the same dataset but it fits in where the weights of the incorrectly classified instances are adjusted. SVM is a set of supervised learning methods that can be used for classifying and regression.

12.6 Application of the Proposed Approach in an Experimental Case

12.6.1 EEG Dataset Selection

The dataset was chosen from the available data from BCI Competition IV. It is a motor imagery dataset that contains 59 EEG channels recorded in around 2,000,000 frames. It contains the values of each EEG signal through time, the voltage between the EEG channel and the ground electrode. The recordings were done for 7 subjects. It is a raw dataset with a sampling rate of 1000 Hz. Firstly, it was in a MATLAB format (.mat), using MATLAB computational environment it was converted to CVS format.

For each one of the subjects, two classes of motor imagery were selected. The options were: left hand, right hand, left foot, and right foot. Therefore, there were three options to be classified (rest, class number one, and class number two). With this way of classifying, it has the same number of classes as the most common way of control a robotic hand, once EEG also considers three classes of control (a contraction of muscle number one, contraction of muscle number two and rest).

12.6.2 Data Pre-processing Applied in the Experimental Case

Although it has been previously band-pass filtered between 0.5 and 100 Hz the digitalized to 1000 Hz, the dataset still provides noise and some not useful information. Figure 12.4 shows the raw data from the EEG acquisition. In this figure, just 20 channels of 59 channels are showed to illustrate the solution. Additionally, as can be seen in Fig. 12.4, it is full of noise and does not represent the desired data for analysis.

The study of EEG signals requires a reference-independent measure of the potential field (Junghöfer 1999). Electrode recordings measure the electrical difference between two points and express them in micro-Volts. One of those two points is the ground electrode, an electrode connected to the ground circuit of the amplifier. Therefore, any electrode signal that can be displayed have as a reference to the so-called ground electrode, this electrode can pick up noise that does not influence the

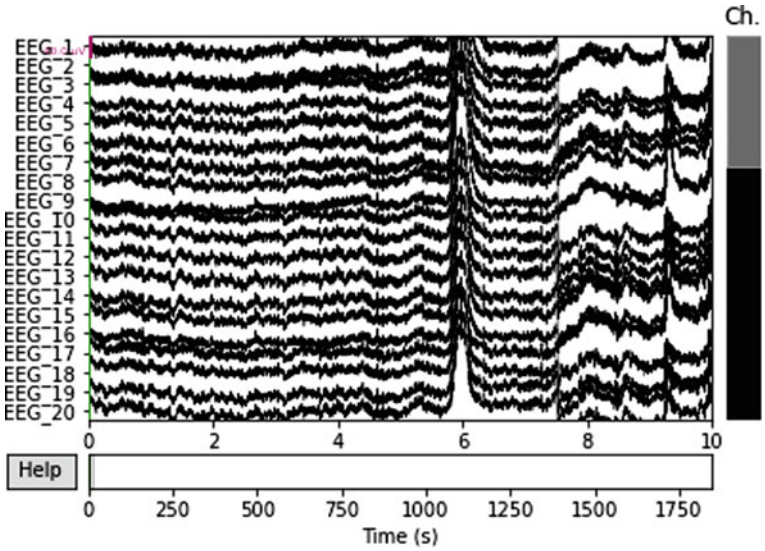

Fig. 12.4 EEG raw data from the 20 more relevant channels

other electrodes (Leuchs 2019). One solution for cleaning the signals from this noise is re-referencing the data.

There are different ways of re-referencing EEG data, and it influences the amplitude of each channel and time point. It redefines the level of the zero voltage and makes all the other EEG signals and channels are related to it (Leuchs 2019).

Average reference is the result of the subtraction of all the EEG signals and the average of all EEG signals through time. It was used here to eliminate part of the noise and can be seen in Fig. 12.5. The figure is the same data as the previous image, re-referenced, with less noise. When using the average re-reference, the amplitudes are overall reduced, and each channel contributes equally to the new reference.

Motor imagery has its power in the alpha (8–12 Hz) and beta (13–30 Hz) frequency bands (Di Nota 2017). To obtain this band a band-pass filter was applied between 7 and 30 Hz. The reduction from the same channels displayed in Figs. 12.4 and 12.5 can be checked in Fig. 12.6, already band-pass filtered.

12.6.3 PCA Application in the Experimental Case

With 59 electrodes placed on the subject's scalp, some not valuable information for classification can also be recorded. This step aims to reduce the number of EEG channels that are going to be analyzed and classified by the neural network.

For accomplishing this step, a matrix was made for each subject with the dimension (frames x channels). For applying PCA, the data needs to be standardized. It

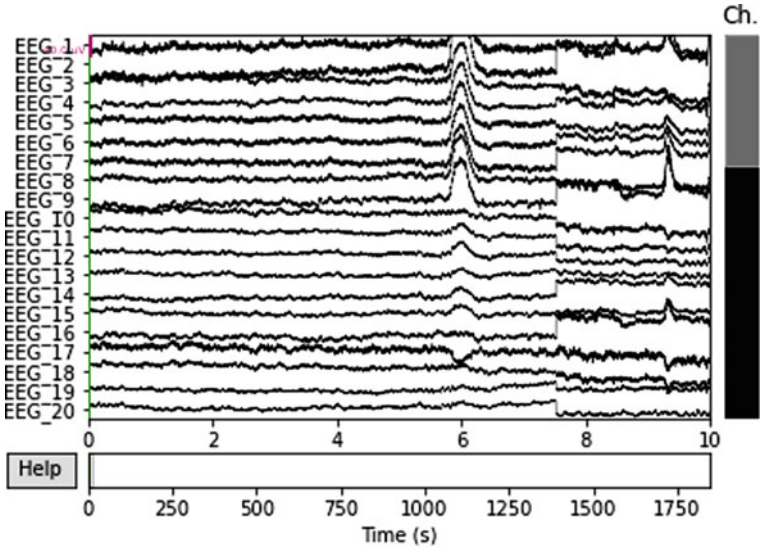

Fig. 12.5 Re-referenced data of EEG raw data

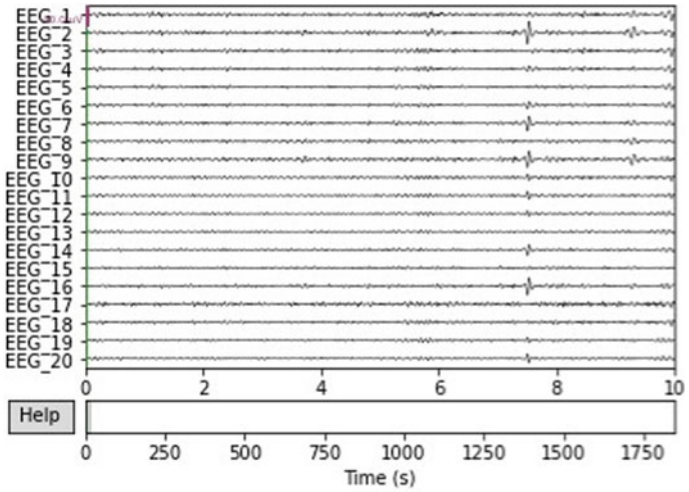

Fig. 12.6 EEG signal filtered (7–30 Hz band-pass filtered)

is done by applying Eq. (12.1) to the dataset. After obtaining the standardized data, a covariance matrix is calculated using Eq. (12.5) for every possible relationship in the dataset. From this covariance matrix, the eigenvectors and eigenvalues were extracted. The eigenvectors were organized in a decreasing form in a matrix according

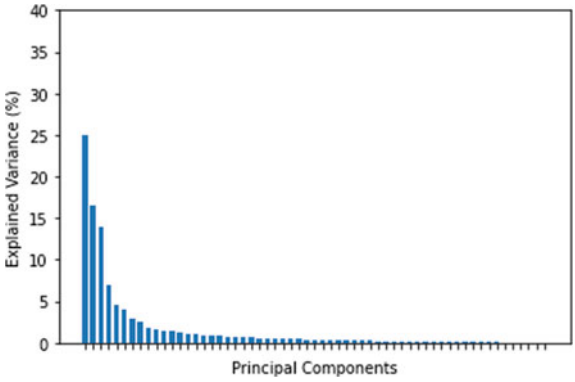

Fig. 12.7 Explained variance of the data

to the value of its respective eigenvalue. These eigenvectors are the principal components. They are the components that explain the biggest part of the variance in the dataset.

For each subject's EEG data that was going to be analyzed, it was verified how many principal components were needed to explain the biggest part of the data, in this case, the number of principal components needed to explain at least 75% of the data's variance. Therefore, the number of channels used as an input for the model was equal to the number of necessary channels to explain the desired percentage of the variance. Based on this, Fig. 12.7 shows a bar plot of all the principal components of the total data. Each of the components has its importance on the data variance and they are ordered by the highest to the lowest considering their percentage on the representativeness of the data's total variance.

As can be observed in Fig. 12.7, some components contribute to a little percentage of the overall variance. Those principal components can be discarded as they would not have a big influence on the final data analysis. With the usage of only 10 of the first PCs together, it is possible to explain 79.67% of the data, as shown in Fig. 12.8. Therefore, the dimension of the dataset could be reduced for those 10 principal components that explain the variance of the biggest part of the data. Instead of analyzing 59 EEG channels, only 10 PCs are going to be analyzed.

Once the dimensionality is reduced by selecting the meaningful principal components that explain the biggest part of the data's variance, the chosen principal components are used in the classification step. Now, with less information to analyze, a more accurate analysis can be done with less computational time.

12.6.4 Data Classification with LSTM Classifier

In this part, the data is classified and tested to check the accuracy of the approach. The PCs were chosen to be used as an input for the RNN classification method. LSTM is an RNN used for time series, and, for its design, the hyperparameters of the

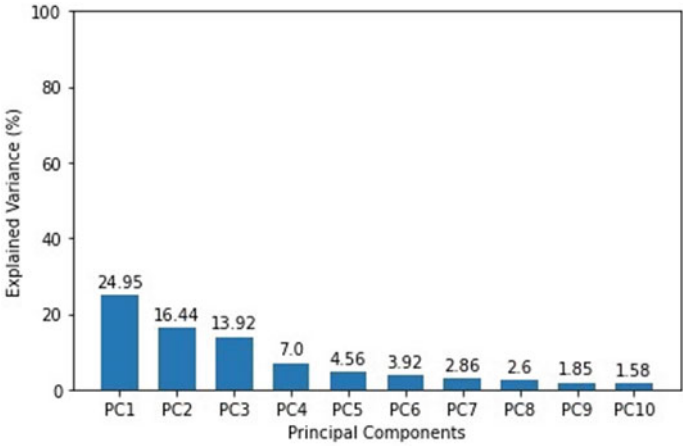

Fig. 12.8 Explained variance of the selected principal components

RNN need to be selected. The number of layers, the number of neurons per layer, the chosen optimizer, the batch size used for the training, and the number of epochs is the parameters to be considered and tested to have a more tuned LSTM. For obtaining better accuracy, different configurations were tested.

A first run was done for choosing the number of layers that would better fit the problem. Different LSTM configurations were tried with a batch size of 32 and 5 epochs as can be seen in Table 12.1. Firstly, a configuration of 20 neurons per layer, using Adam (Adaptive Moment Estimation) optimizer was performed 3 times changing the number of layers each time. It was checked that the LSTM 1 outperformed the other 2 having an accuracy of 84.43% and a mean squared error of 0.11 while the LSTM 2 had an accuracy of 83.43 with a mean squared error of 0.11 and the worst perfume of this first trial was from the LSTM 3 that had an accuracy of 79.54% and a mean squared error of 0.14. With the results shown in Table 12.2,

Table 12.1 Configurations of the first test of LSTMs tuning according to the number of layers

LSTM	1	2	3
Number of layers	3	2	1
Number of neurons per layer	20	20	20
Optimizer	Adam	Adam	Adam
Drop out	0.2	0.2	0.2

Table 12.2 Accuracy and mean squared error from the first LSTM configurations for tuning

LSTM configuration	Accuracy (%)	Mean squared error
1	84.43	0.11
2	83.43	0.11
3	79.54	0.14

Table 12.3 Configuration of the second test of LSTMs tuning according to the number of layers

LSTM	4	5	6	7
Number of layers	1	1	1	1
Number of neurons per layer	1	10	20	30
Optimizer	Adam	Adam	Adam	Adam
Drop out	0.2	0.2	0.2	0.2

the second set of tests was done to find the number of neurons per layer that would be the best fit for the LSTM.

In the second run for finding the best configuration, the aim was to find the number of neurons per layer. They were tested with the same batch size and the same epoch that before, the number of layers equal to one for saving computational time, Adam optimizer was used and a drop out of 0.2. As is shown in Table 12.3, the number of neurons per layer tested were 1, 10, 20, and 30, more than this would take too much computational time.

The second run for tuning the LSTM checked that the highest number of neurons obtained the best accuracy. The LSTM 7 outperformed the LSTMs 4, 5, and 6 with an accuracy of 85.27% while the others achieved the accuracies of 66.67%, 73.67%, and 79.74%, respectively. The mean squared error calculated from the LSTM performance was also better for the last one of them. With a mean squared error of 0.10, the LSTM with the setup #7 had a better result than the other that had the root mean squared errors higher than 0.14 as shown in Table 12.4.

The last run was done to find the number of epochs that best fits the problem. Three tests were done with 1 layer, one neuron per layer, Adam optimizer, and 5 training epochs, using the batch sizes of 32, 128, and 1024, as can be seen in Table 12.5. The results are in Table 12.6.

Table 12.4 Accuracy and mean squared error resulting from the second LSTM configurations for tuning

LSTM configuration	Accuracy (%)	Mean squared error
4	66.67	0.20
5	73.67	0.17
6	79.74	0.14
7	85.27	0.10

Table 12.5 Configurations of the third test of LSTMs tuning according to the number of layers

LSTM	8	9	10
Number of layers	1	1	1
Number of neurons per layer	1	1	1
Optimizer	Adam	Adam	Adam
Drop out	0.2	0.2	0.2
Batch size	32	128	1024

Table 12.6 Accuracy and mean squared error resulting from the third LSTM configurations for tuning

LSTM configuration	Accuracy (%)	Mean squared error
8	66.67	0.20
9	66.70	0.20
10	66.66	0.20

With the results from Table 12.6, the LSTM chosen configuration was an LSTM with 3 layers, 30 neurons per layer, using Adam optimizer a batch size of 128. Another approach was done for classifying the data. After preprocessing the data where 473 epochs were extracted from the dataset. The time point where the subject started one movement, or another was detected and after this time point, 1,000 frames were stored. From this new dataset, 7 features were extracted from each one of the epochs, features as mean, variance, skewness, kurtosis, peak-to-pick distance, and maximum peak-to-peak distance. And those filters were used to classify each epoch. The same LSTM previously found was used in this method.

To sum up, two methods were done, one directly analyzing the continuous signal (Method 1) and another one extracting 8 features from the signal and analyzing the feature vector made from it (Method 2).

12.6.5 The Results of the PCA + LSTM-RNN Approach Application in an Experimental Case

From the tests previously done for choosing the LSTM configuration, in Sect. 12.6.4, an LSTM with 3 layers, 30 neurons per layer, Adam optimizer, 30 epochs and a batch size of 128 was chosen to be the classifier to solve the problem of this research, as shown on Table 12.7.

This configuration was validated with fivefold cross-validation and showed a mean accuracy of 94.41% ± 0.23%. LSTM has its focus on predictions, for using it as a classifier, the numbers it predicted need to be rounded. Once was done, the method's accuracy decreased a bit, and the result was accuracy of 91.05%.

With the algorithm already trained, the model was used to classify a dataset with 370,015 frames and was used to classify each one of those frames, the chosen

Table 12.7 Configuration of the chosen LSTM

Number of layers	3
Drop out	0.2
Neurons per layer	20
Optimizer	Adam
Epochs	30
Batch Size	128

Table 12.8 LSTM Confusion matrix

88,883	6955	1412
9065	166,336	4123
4392	7146	81,703

Table 12.9 Accuracy results from the classification

Machine learning method	Accuracy (%)
LSTM	91.05
SVM	48.21
Random forest	48.46
Adaboost	51.05

configuration was able to classify the dataset with an accuracy of 91.05% of the re-referenced dataset after the principal components had been applied. The confusion matrix shows in its principal diagonal the numbers of the right classifications of the data, it shows that in 336,922 frames, the algorithm was classified correctly (Table 12.8).

To do the comparison of the LSTM accuracy, some machine learning classifiers were used. The related works (Jia et al. 2020; Al-Saegh et al. 2021; Rajapriya et al. 2021; Lin et al. 2021) suggest the methods such as SVM, Random Forest, Adaboost. With the chosen approach, LSTM is the best choice for classifying the data. It outperformed the other methods classifying the data while all the other methods, had their accuracies below 52% (Table 12.9).

For the second approach chosen, where the 7 features were extracted from each one of the epochs after the data had been preprocessed and the dimension had been reduced, the classifiers have shown a low accuracy for classifying the three different classes. LDA showed an accuracy of 53.68%, Random Forest was the best one of the machine learning models that showed an accuracy of 57.89%, AdaBoost had an accuracy of 54.73% and SVM, being the second-best one of the machine learning models, showed an accuracy of 56.84%. With the same approach, LSTM had an accuracy of 63.68% outperforming the other methods (Table 12.10).

Table 12.10 Comparison of the two methods used

Method 1		Method 2	
LSTM	91.05%	LSTM	63.68%
SVM	48.21%	SVM	56.84%
Random forest	48.46%	Random forest	57.89%
Adaboost	51.05%	Adaboost	54.73%
LDA	48.40%	LDA	53.68%

12.7 Conclusion and Future Research

This chapter explored an approach which it integrates PCA and LSTM-RNN approaches to evaluate the performance to control a prosthetic hand based on the signals generated from the user's intention of movement recorded from EEG. The approach proposed in composed of 4 parts: (i) EEG Dataset Acquisition, (ii) Data Pre-Processing, (iii) PCA, and (iv) LSTM Classifier.

Principal component analysis has appeared to be a good approach for this problem, reducing the dimension to be analyzed for only 10 principal components, it performed a dimensionality reduction of 83.05% of the total number of channels to be analyzed and allowed the classification of more meaningful data. LSTM has shown promising results with a mean accuracy of 91.05% on method 1 and 63.08% on method 2, outperforming all the other methods.

The objective of designing a suitable BCI application was not totally accomplished using dimensionality reduction with PCA combined and LSTM-RNN. However, with the results is possible to conclude that the approach of using principal component analysis with a recurrent neural network is an accurate approach for classifying the data. An EEG dataset was well classified in three classes (same number of classes used in a myoelectric hand control), therefore, it is possible to use electroencephalography on the robotic hand's control. On the other hand, for a BCI application, where the processing needs to be done online, the method chosen in this project may not fit its due to its need for a time series to work. BCI applications need real-time data processing, thus, a classical machine learning approach could be a better choice, a more tuned machine learning classifier and a different approach of data reduction could make a better performance.

For the future perspectives for this project would be trying to outperform the LSTM classification with a machine learning approach. Once a promising result of a machine learning approach is found, develop a BCI system, and applied on the control of a robotic hand for checking its real accuracy.

Acknowledgements The authors would like to thank the Pontifical Catholic University of Parana (PUCPR) for financial support to the development of this research.

References

Al-Saegh A, Dawwd SA, Abdul-Jabbar JM (2021) Deep learning for motor imagery EEG-based classification: a review. Biomed Signal Process Control 63:102172. https://doi.org/10.1016/j.bspc.2020.102172

Bäckström M, Tidare J (2016) A brain-actuated robot controller for intuitive and reliable manoeuvring. Master thesis, Mälardalen University

Badawy A, Alfred R (2020) Myoelectric prosthetic hand with a proprioceptive feedback system. J King Saud Univ Eng Sci 32(6):388–395. https://doi.org/10.1016/j.jksues.2019.05.002

Bianchi FM, Maiorino E, Kampffmeyer MC, Rizzi A, Jenssen R (2017) Recurrent neural network architectures. In: Bianchi FM, Maiorino E, Kampffmeyer MC, Rizzi A, Jenssen R (eds) Recurrent

neural networks for short-term load forecasting: an overview and comparative analysis. Springer International Publishing, Cham, pp 23–29

Canciglieri MB, Leite AFCS de M, Szejka AL, Júnior OC (2019) An approach for dental prosthesis design and manufacturing through rapid manufacturing technologies. Int J Comput Integr Manuf, 0(0):1–16. https://doi.org/10.1080/0951192X.2019.1636410

Corchs S, Chioma G, Dondi R, Gasparini F, Manzoni S, Markowska-Kaczmar U, Mauri G, Zoppis I, Morreale A (2019) Computational methods for resting-state EEG of patients with disorders of consciousness. Front Neurosci 13:807. https://doi.org/10.3389/fnins.2019.00807

Cordella F, Ciancio AL, Sacchetti R, Davalli A, Cutti AG, Guglielmelli E, Zollo L (2016) Literature review on needs of upper limb prosthesis users. Front Neurosci 10:209. https://doi.org/10.3389/fnins.2016.00209

dos Santos VP, Silveira DR, Caffaro RA (2006) Risk factors for primary major amputation in diabetic patients. Sao Paulo Med J 124:66–70. https://doi.org/10.1590/S1516-31802006000200004

Eisenberg GD, Fyvie KGHM, Mohamed A-K (2017) Real-time segmentation and feature extraction of electromyography: towards control of a prosthetic hand. IFAC-Pap 50(2):151–156. https://doi.org/10.1016/j.ifacol.2017.12.028

Gannouni S, Aledaily A, Belwafi K, Aboalsamh H (2021) Emotion detection using electroencephalography signals and a zero-time windowing-based epoch estimation and relevant electrode identification. Sci Rep 11(1):7071. https://doi.org/10.1038/s41598-021-86345-5

Geiss LS, Li Y, Hora I, Albright A, Rolka D, Gregg EW (2019) Resurgence of diabetes-related nontraumatic lower-extremity amputation in the young and middle-aged adult U.S. population. Diabetes Care 42(1):50–54. https://doi.org/10.2337/dc18-1380

Graham EM, Hendrycks R, Baschuk CM, Atkins DJ, Keizer L, Duncan CM, Mendenhall SD (2021) Restoring form and function to the partial hand amputee: prosthetic options from the fingertip to the palm. Hand Clin 37(1):167–187. https://doi.org/10.1016/j.hcl.2020.09.013

Graimann B, Allison B, Pfurtscheller G (eds) (2010) Brain-computer interfaces: revolutionizing human-computer interaction. Springer, Berlin Heidelberg

Hiriyannaiah S, Srinivas AMD, Shetty GK, Siddesh GM, Srinivasa KG (2020) Chapter 4—a computationally intelligent agent for detecting fake news using generative adversarial networks. In: Bhattacharyya S, Snášel V, Gupta D, Khanna A (eds) Hybrid computational intelligence. Academic Press, pp 69–96

Hoffstad O, Mitra N, Walsh J, Margolis DJ (2015) Diabetes, lower-extremity amputation, and death. Diabetes Care 38(10):1852–1857. https://doi.org/10.2337/dc15-0536

Iftikhar M, Khan SA, Hassan A (2018) A survey of deep learning and traditional approaches for EEG signal processing and classification. In: 2018 IEEE 9th annual information technology, electronics and mobile communication conference (IEMCON). pp 395–400

Jia G, Lam H-K, Liao J, Wang R (2020) Classification of electromyographic hand gesture signals using machine learning techniques. Neurocomputing 401:236–248. https://doi.org/10.1016/j.neucom.2020.03.009

Kashef SR, Amini S, Akbarzadeh A (2020) Robotic hand: a review on linkage-driven finger mechanisms of prosthetic hands and evaluation of the performance criteria. Mech Mach Theory 145:103677. https://doi.org/10.1016/j.mechmachtheory.2019.103677

Keyl P, Schneiders M, Schuld C, Franz S, Hommelsen M, Weidner N, Rupp R (2019) Differences in characteristics of error-related potentials between individuals with spinal cord injury and age- and sex-matched able-bodied controls. Front Neurol 9:1192. https://doi.org/10.3389/fneur.2018.01192

Kropotov JD (2016) Chapter 1.6—event-related potentials. In: Kropotov JD (ed) Functional neuromarkers for psychiatry. Academic Press, San Diego, pp 59–78

Kumar DK, Jelfs B, Sui X, Arjunana SP (2019) Prosthetic hand control: a multidisciplinary review to identify strengths, shortcomings, and the future. Biomed Signal Process Control 53:101588. https://doi.org/10.1016/j.bspc.2019.101588

Laclé A, Valero-Juan LF (2012) Diabetes-related lower-extremity amputation incidence and risk factors: a prospective seven-year study in Costa Rica. Rev Panam Salud Pública 32:192–198. https://doi.org/10.1590/S1020-49892012000900004

Lekshmi SS, Selvam V, Pallikonda Rajasekaran M (2014) EEG signal classification using principal component analysis and wavelet transform with neural network. IEEE, Melmaruvathur, India, pp 687–690

Li X, Samuel OW, Zhang X, Wang H, Fang P, Li G (2017) A motion-classification strategy based on sEMG-EEG signal combination for upper-limb amputees. J NeuroEngineering Rehabil 14(1):2. https://doi.org/10.1186/s12984-016-0212-z

Lin L, Li M, Ma L, Baziar A, Ali ZM (2021) Hybrid RNN-LSTM deep learning model applied to a fuzzy based wind turbine data uncertainty quantization method. Ad Hoc Netw 123:102658. https://doi.org/10.1016/j.adhoc.2021.102658

Machado S, Cunha M, Velasques B, Minc D, Bastos VH, Budde H, Cagy M, Piedade R, Ribeiro P (2009) Interface cérebro-computador: novas perspectivas para a reabilitação. Rev Neurociências 17(4):329–335. https://doi.org/10.34024/rnc.2009.v17.8525

Modaresi F, Araghinejad S, Ebrahimi K (2018) A comparative assessment of artificial neural network, generalized regression neural network, least-square support vector regression, and K-nearest neighbor regression for monthly streamflow forecasting in linear and nonlinear conditions. Water Resour Manag 32(1):243–258. https://doi.org/10.1007/s11269-017-1807-2

Nunez PL, Srinivasan R (2006) Electric fields of the brain: the neurophysics of EEG, 2nd edn. Oxford University Press, New York

Nuwer MR, Coutin-Churchman P (2014) Brain mapping and quantitative electroencephalogram. In: Aminoff MJ, Daroff RB (eds) Encyclopedia of the neurological sciences, 2nd edn. Academic Press, Oxford, pp 499–504

Rajapriya R, Rajeswari K, Thiruvengadam SJ (2021) Deep learning and machine learning techniques to improve hand movement classification in myoelectric control system. Biocybern Biomed Eng 41(2):554–571. https://doi.org/10.1016/j.bbe.2021.03.006

Reimers C, Requena-Mesa C (2020) Chapter 13—deep learning—an opportunity and a challenge for geo- and astrophysics. In: Škoda P, Adam F (eds) Knowledge discovery in big data from astronomy and earth observation. Elsevier, pp 251–265

Saint-Elme E, Larrier M, Kracinovich C, Renshaw D, Troy K, Popovic M (2017) Design of a biologically accurate prosthetic hand. In: 2017 International symposium on wearable robotics and rehabilitation (WeRob), pp 1–2

Salelkar S, Ray S (2020) Interaction between steady-state visually evoked potentials at nearby flicker frequencies. Sci Rep 10(1):5344. https://doi.org/10.1038/s41598-020-62180-y

Silva R, Rudek M, Szejka AL, Canciglieri Junior O (2018) Machine vision systems for industrial quality control inspections. In: Product lifecycle management to support industry 4.0. Springer Nature Switzerland, Turin, pp 631–641

Sovierzoski MA, Argoud FIM, de Azevedo FM (2008) Identifying eye blinks in EEG signal analysis. In: 2008 International conference on information technology and applications in biomedicine, pp 406–409

Srinivasan R, Nunez PL (2012) Electroencephalography. In: Ramachandran VS (ed) Encyclopedia of human behavior, 2nd edn. Academic Press, San Diego, pp 15–23

Subasi A, Ismail Gursoy M (2010) EEG signal classification using PCA, ICA, LDA and support vector machines. Expert Syst Appl 37(12):8659–8666. https://doi.org/10.1016/j.eswa.2010.06.065

Tharwat A (2016) Principal component analysis—a tutorial. Int J Appl Pattern Recognit 3(3):197–240. https://doi.org/10.1504/IJAPR.2016.079733

Vaskov AK, Chestek CA (2021) Brain-machine interfaces: lessons for prosthetic hand control. Hand Clin 37(3):391–399. https://doi.org/10.1016/j.hcl.2021.04.003

Yang D, Gu Y, Thakor NV, Liu H (2019) Improving the functionality, robustness, and adaptability of myoelectric control for dexterous motion restoration. Exp Brain Res 237(2):291–311. https://doi.org/10.1007/s00221-018-5441-x