VRIJE
UNIVERSITEIT
BRUSSEL

MUSCLE CLASSIFICATION VIA HYBRID CNN-LSTM ARCHITECTURE AND SEMG SIGNALS

A thesis presented for the degree of Master of Science in Applied Computer Science

Esteban Velásquez Rendón

September 2022

Promotors and Supervisors:
prof. dr. Bart JANSEN
prof. dr. Lubos OMELINA
prof. dr. Jan CORNELIS

Department: ETRO

Chapter 1

Abstract

Electromyography is a technique for examining muscle activity during movement as it can provide details on the timing and degree of muscle activation. During muscle contraction, electrical activity is produced and can be detected through electrodes, information that can later be addressed for a wide variety of applications, including biomedical applications (like controlling an artificial limb), VR hand gesture implementations, human-computer interaction; or it can also be used for a clinical diagnosis like in the assessment and rehabilitation of neuromuscular disorders or sports injuries. This technique is either conducted invasively using fine needles (intramuscular) at the expense of the patient's discomfort, or it can be also carried out using non-invasive surface electrodes. Surface EMG has the advantage of being easy to use and painless, which makes it ideal for wearable technology.

The information concerning the specific muscle from which the sEMG technique is being applied is often required for traceability and signal analysis purposes. Yet, this information is typically provided manually by the user/practitioner, which translates into additional time and makes the procedure more prone to human error. The purpose of this thesis is to evaluate the performance of a CNN-LSTM model to correctly classify the muscle directly from the EMG signal. In this sense, muscle contraction bursts are provided to the model as inputs. The hybrid CNN-LSTM approach generally allows for effective feature extraction and the ability to learn short-term and long-term sequential dependencies. Two different training setups are proposed: one using weight initialization provided from layer-wise unsupervised pretraining strategy and the other one using random initialization. Morever two validation scenarios are as well proposed to assess the performance of the model: in the first one, the model is tested on new contraction bursts from already seen subjects in the training step (obtaining 95% and 92% accuracy for the two training setups respectively), whereas in the second one, one subject is left out of the training, and the whole testing is done on those samples (obtaining 48% accuracy on both training setups).

Contents

Chapter 2

Introduction

Electromyography (EMG) has been used in medicine since the late nineteenth century to study the electrical activity of muscles [23]. A subtype of EMG, called Surface EMG (sEMG), is a nowadays growing technique that captures the electrical activity that is transferred to the surface of the skin, where non-invasive electrodes are located. This technique is currently employed for different types of applications, such as rehabilitation, advanced myoelectric control of prosthetic systems, gesture recognition, training program effectiveness, ergonomics, and movement analysis [30].

Surface EMG can be seen as an easier-to-use alternative for patients/users as it does not harm the soft tissue nor leave subsequent scarring, compared to other invasive EMG approaches. Also, this technique allows for unlimited reiteration of tests in the very same spot/muscle, making it ideal for the rehabilitation of neuromuscular disorders. Throughout an electromyography study, physicians and practitioners keep track of the muscles where sEMG sensors are placed, either to assess the overall performance of the muscle or for periodic evaluation in the rehabilitation process. The traceability of the muscle involved is crucial and yet, it is currently done manually, making room for human errors and latency in the process.

In the present study, sEMG is employed to record electrical activity over different muscles of interest with the goal of capturing different muscle contraction bursts, from where a hybrid deep learning model can analyze patterns (muscle contraction signature) and infer near which muscle the sensor is located (via classification). It's important to highlight that while muscles contract, the recorded sEMG contraction bursts are subject to anatomical and physiological factors that influence the signal such as muscle size, amount of motor unit potentials, demanded force, muscle fatigue, presence of abnormal behavior (fibrillation, fasciculation), existing conditions (myopathy or neuropathy), electrical interference, among others. All these factors increase the complexity of muscle recognition tasks from sEMG signals, making deep learning models, such as the proposed hybrid CNN-LSTM, an appropriate candidate solution considering its capability of extracting multiple relevant features effectively, while also involving the time-series dependency inherent in sEMG recordings.

Chapter 3

Objectives

The goal of this thesis is to study the performance of a supervised deep learning model in the task of classifying muscles from EMG data, more precisely from muscle contraction bursts. To achieve it, specific sub-objectives are defined:

- Formulate a data collection protocol where different muscle activations can be recorded and correctly labeled.
- Explore state-of-the-art machine learning models and deep learning models used over sEMG data.
- Analyze and process the sEMG data obtained from the data collection protocol.
- Implement the deep learning model for the muscle multiclass classification task.
- Train and evaluate the performance of the model in two main scenarios: one assessing the performance of the model to correctly classify new contraction bursts from previously seen subjects, and a second scenario evaluating the model on new subjects that are not involved in the training.

Chapter 4

Context

4.1 Muscle Activation

Muscle activation can be defined as the muscle's ability to contract and produce force. This ability is what enables the human body's movement when in coordination with the brain. The brain transmits excitation signals through the Central Nervous System (CNS) whenever the muscles of the body need to be engaged for a certain task. For this purpose, the CNS uses receptors to collect information from the exterior (environment) and the inner body. High-level areas in the brain then combine this vast amount of information with prior knowledge to produce appropriate motor responses [16]. The nervous and muscular systems are connected via motor units, which are a specific group of muscle fibers controlled by a single neuron. The motor unit structure can be seen in Figure 4.1. It's important to highlight that the number of motor units in each muscle varies, and so does the average number of muscle fibers innervated by the motor neurons. The ratio of innervation is an indicator of the average number of muscle fibers that will respond to a single motor neuron's action potential. As Roger M. Enoka [15] mentions, motor unit behavior varies substantially during voluntary contractions and this variability is present due to the differences in motor-unit morphology. Moreover, as the recruitment of motor units increases, the muscle force is increased as well, starting from small motor units to larger ones. The electrical activity in the form of action potentials propagates along the muscle fibers when a muscle contracts.

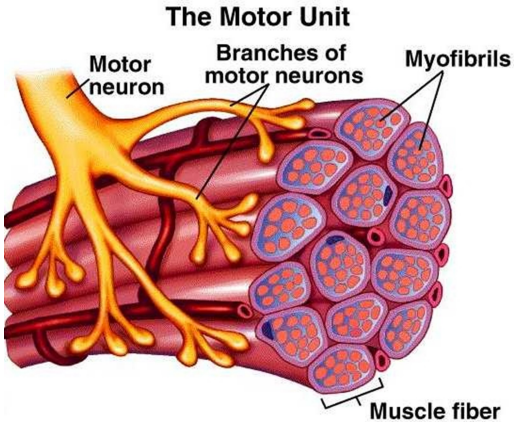


Figure 4.1: Motor Unit Structure [43]

4.2 Electromyography

Electromyography is a diagnostic procedure that measures the electrical activity of skeletal muscles. The result can help identify the activation level, recruitment order, as well as different types of abnormalities that directly concern neuromuscular dysfunction and problems involving the transmission of the signals from the brain down to the nerves in the arms and feet. This diagnostic procedure serves as an important tool for physicians to determine what treatment is required in case of disease or injury.

To perform electromyography, an electromyograph is used to capture and monitor the electric potential generated by muscle cells. This electric potential is introduced by minor electrical currents produced during the exchange of ions across muscle membranes when there is muscle activity, and it is measured through electrodes [31]. The electrodes can be categorized into two main classes: inserted (such as needle and fine wire electrodes) and surface electrodes. Surface electrodes are used in a subtype of electromyography called Surface Electromyography (sEMG) and it's the technique employed in the present thesis.

In addition to this, some sEMG devices make use of multiple adjacent electrodes arranged in a two-dimensional grid configuration. This subtype of sEMG is commonly referred to as High-Density Surface Electromyography (HD-sEMG), which is a method for the recording of Motor Unit Action Potentials (MUAP) over a muscle, using an established grid configuration of electrodes. These devices can provide both: spatial and temporal characteristics of the signal, in contrast to conventional surface electromyography (sEMG), enabling a more thorough evaluation of muscle electrical activity. This technique has received significant attention for applications like signal decomposition, analysis of changes in the spatial configuration of motor unit potentials and movement intention identification [38].

When analyzing a sEMG recording, one can commonly evidence two main behaviors: the rest period, where the muscle is in a relaxed state and the electrical activity does not deviate too much from the baseline; and the contraction burst period, where few or multiple motor units activate different muscle fibers, causing an increment in the signal amplitude and frequency. Note that

every muscle provides a different electromyogram during an active contraction burst. Muscle size and amount of motor units potentials, demanded force, muscle fatigue, presence of abnormal behavior (fibrillation, fasciculation), existing conditions (myopathy or neuropathy), electrical interference, among others, are some of the factors that directly influence EMG recordings [24, 36] and that must be taken into account when processing these signals.

EMG signal processing by means of powerful and advanced approaches is becoming a critical requirement, especially for sEMG, where there are still limitations linked to EMG detection and analysis techniques. In this sense, artificial intelligence (AI) and mathematical techniques are breaking new ground in the subject: AI models such as Artificial Neural Networks (ANN), Dynamic recurrent Neural Networks (DRNN), Genetic Algorithms (GA), among other machine learning techniques are being implemented for this purpose. Likewise, mathematical techniques such as Wavelet transform, time-frequency approaches, Fourier transform and other statistical approaches are addressing this matter [37].

4.3 Machine Learning and Deep Learning Models

Machine learning is a subset of Artificial Intelligence (AI) that refers to computer algorithms that can categorize, find patterns, and make predictions based on data. It involves the process of learning from the data and the posterior generation of an output or outputs linked to the goal of the algorithm. In this process, the machine learns by itself without being explicitly programmed; this is achieved by using a variety of techniques to address different data goals. Two main types of learning are normally employed in machine learning models: supervised and unsupervised learning. In the first one, input-label pairs are given during the training step of the model with the aim that the model adapts the weights that define the mapping between the inputs and the outputs, and after the learning step is done, a subsequent testing phase on unseen samples can be performed to validate the model performance. Supervised learning is commonly used for classification or regression tasks. The second type of learning, unsupervised learning, relies only on the input, so the model ends up learning the structure of the data, which is useful for applications that involve clustering and feature reduction [29].

In contrast to traditional algorithms that require handcrafted features, deep learning techniques can learn representations of data that can be used to solve complicated problems directly from the data. Deep learning models use a layered architecture that incorporates artificial neural networks (ANN) that behave along similar principle as the human brain network [20] and that is far more complex, but also more powerful than traditional machine learning models. In general, deep learning algorithms require minimal human intervention due to autonomous feature engineering. However, the required amount of data and the computing power that these models demand is often considerable and thus needs to be taken into account when choosing these models. These models have had a lot of success for applications such as pattern recognition, natural language processing (NLP), speech recognition, and many others [26].

There are different types of deep neural networks. The three main types used in the present thesis are convolutional neural networks (CNN), long-short-term memory (LSTM), and autoencoder (AE). These three are further explained in the following sections.

4.3.1 Convolutional Neural Networks (CNN)

CNN is a type of ANN used to process structured arrays. They are widely used in applications involving image processing (such as image classification) but as well in text for natural language processing (for text classification for instance) and computer vision applications. This type of network typically employs the following type of layers: the convolutional layer, the nonlinearity layer, the pooling layer and the fully connected layer [2].

- The convolutional layer, in which a mathematical linear operation called convolution is carried out (matricial operation). In this operation, hyperparameters such as the number of filters, the kernel size and the stride are defined. Successive convolutions over the input array (or image) in an iterative process allows the model to extract relevant features from the data by correctly updating weights/parameters for the deep learning goal.
- The nonlinearity layer. This layer can be used to frame the output from the previous layer by applying a particular activation function. In this sense, a non-linear activation function such as sigmoid, hyperbolic tangent, ReLU or LeakyReLU can be employed. Figure 4.2 displays these activation functions. Notice that ReLU stands for Rectified Linear Unity, and it transforms negative values to zero and keeps only the positive ones; while in Leaky ReLU a slight negative slope is introduced, so that when the input value is less than 0, the function produces small negative outputs. Leaky ReLU is a popular and efficient non-linear activation function that tackles the a problem of vanishing gradients (which can be present in standard ReLu) [12, 21] and is a non-linear activation function employed in the hybrid CNN-LSTM model of the present thesis.

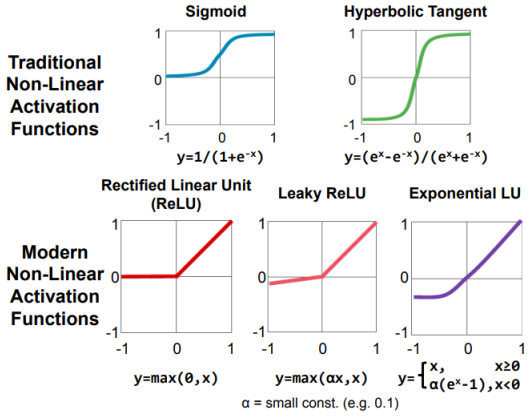


Figure 4.2: Non-Linear Activation Functions [42]

- The pooling layer, where a down-sampling process is performed that reduces the dimensionality of the feature maps obtained in the previous layers. This allows that the captured features are generalized, and lenables the network to recognize the features regardless of the precise location in the input image or input array.
- The fully connected layer, or also called the dense layer, where the output of the previous layer is connected to every activation unit from the next layer.

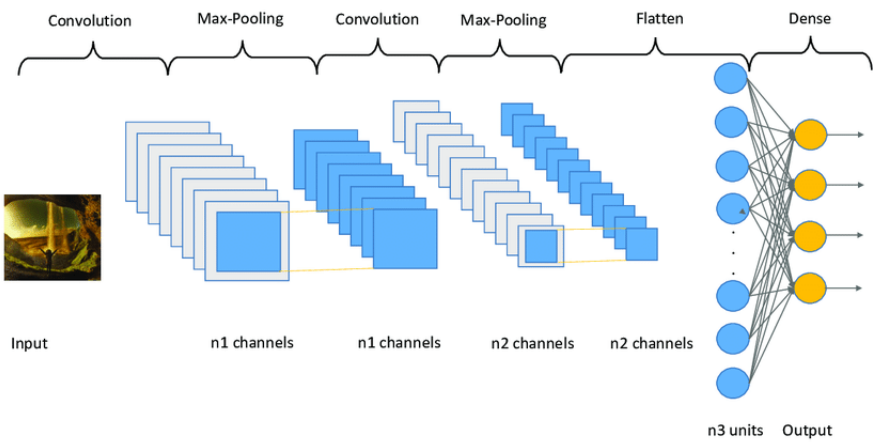


Figure 4.3: Convolutional Neural Network [17]

Figure 4.3 displays a vanilla CNN. Notice that multiple stacks of these layers can be combined together and, when doing so, the feature maps from the first convolutional layers can extract basic features in images, such as lines and edges, while deeper convolutional layers extract more complex features like geometrical shapes or more detailed patterns.

4.3.2 Long Short Term Memory (LSTM)

LSTM is used for time dependent data processing. This type of models can learn and process from the sequence of data and they are being widely used nowadays for applications that involve forecasting, like deep solar radiation forecasting [18], air quality prediction [25], earthquake trend prediction [9], among other.

When the LSTM architecture is being trained, each LSTM block will eventually learn which memories to keep and which to drop for the task goal, meaning that previous events influence the prediction being done by this model. In order to further understand LSTM, it's important to understand Recurrent Neural Networks (RNNs), since LSTMs are a subtype of RNN.

RNNs are a type of neural network in which the output from the previous step is considered as input to the current step (building a relationship between hidden states while processing the next timestep). Therefore, each RNN node has 2 inputs: the current input value X_t and the previous hidden state $a_(t-1)$, and 2 outputs: the prediction h_t and the updated hidden state a_t. Notice that a recurrent neural network can be seen as multiple instances of the same network, when unfolding the loops in the network architecture. Figure 4.4 displays the RNN structure.

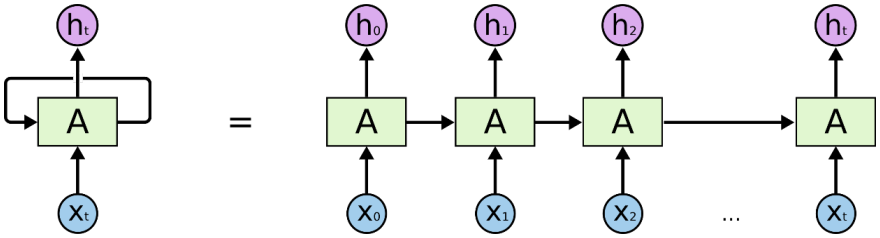


Figure 4.4: RNN Structure [32]

These recurrent neural networks are deeply related with sequences/lists and have been widely used in domains as: speech recognition [4], translation [28] and image captioning [33]. In contrast, LSTMs are, as previously mentioned, a special type of RNN that commonly perform better than traditional RNNs since they not only have a hidden state but also a cell state that allows for longer effective memory [5]. As seen in the previous figure, RNNs connect previous information (or hidden state) as input for the present timestep; however, when there is a big time gap between meaningful previous information and the current timestep they may not perform as required (as the gap increases, RNNs start becoming unable to connect meaningful previous information). This is the main reason LSTMs are then used, to learn long term dependencies. The internal architecture of LSTM is presented in Figure 4.5.

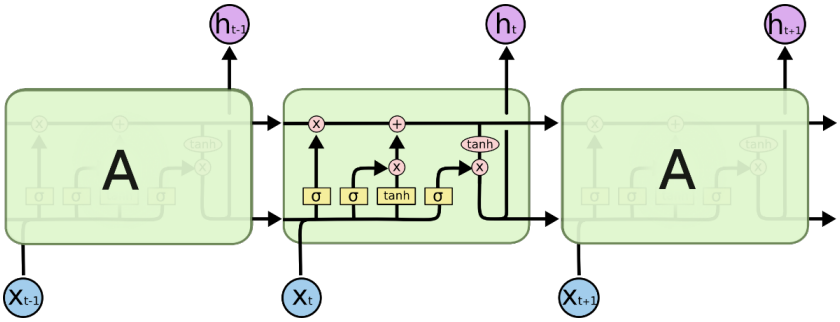


Figure 4.5: LSTM Internal Architecture [32]

Notice that lines in the previous Figures 4.4 and 4.5 represent vectors; the horizontal ones are transmitting a vector from the output of one node to the input of the following one. Operations $\times$ and $+$ represent multiplication and addition. Moreover, different functions like sigma and tanh (shown in the yellow boxes) are also important components of the internal architecture. There are 3 main gates that protect and control the cell state:

- The Forget Gate: In charge of deciding if the previous memory will be forgotten or not based on the previous hidden state, the current input and the associated weights (see first $\times$ from left to right). When the value is 0 after passing through the sigmoid function, then the previous memory is erased (as the previous memory will be multiplied by 0). If the value is 1, then it is kept.

- The Input Gate: In charge of adding a new memory to be kept based on the previous hidden state, the current input and the associated weights (see second $\times$ from left to right); however, this time 2 functions are used: sigmoid and tanh, and these two results are multiplied. The multiplication result is then added to previous memory (if any).
- The Output Gate: In charge of delivering an output/hidden state based on the memory (applying tanh on it) and multiplying it with the result of the sigmoid function applied to the previous hidden state and the input (taking into account the associated weights as well). Notice that in the next timestep the hidden state as well as the memory is carried. This gate is located at the third $\times$ from left to right in the previous figure.

The key feature of LSTM is the long-time memory, which corresponds to the top horizontal line (or bus) in the figure, and this feature is especially important for time series forecasting. It is important to highlight that multiple variants of LSTM exist, but the main structure is the one previously explained.

4.3.3 Autoencoder (AE)

AEs are particular neural networks that learn to transform input data into a latent representation and then reconstruct it back by means of unsupervised learning [7]. These two behaviours are commonly known as encoding and decoding. Figure 4.6 shows an example of an AE architecture. AEs are used for applications such as image denoising [6], feature extraction [27], image generation [40], among others.

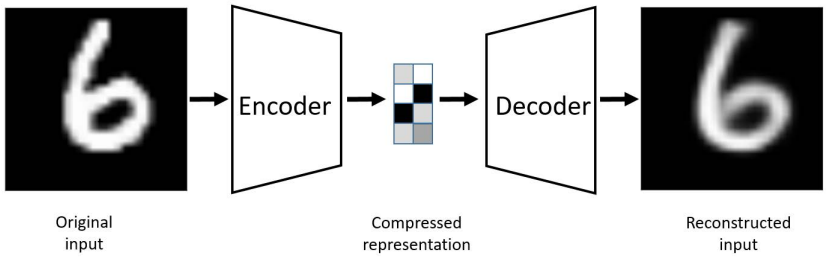
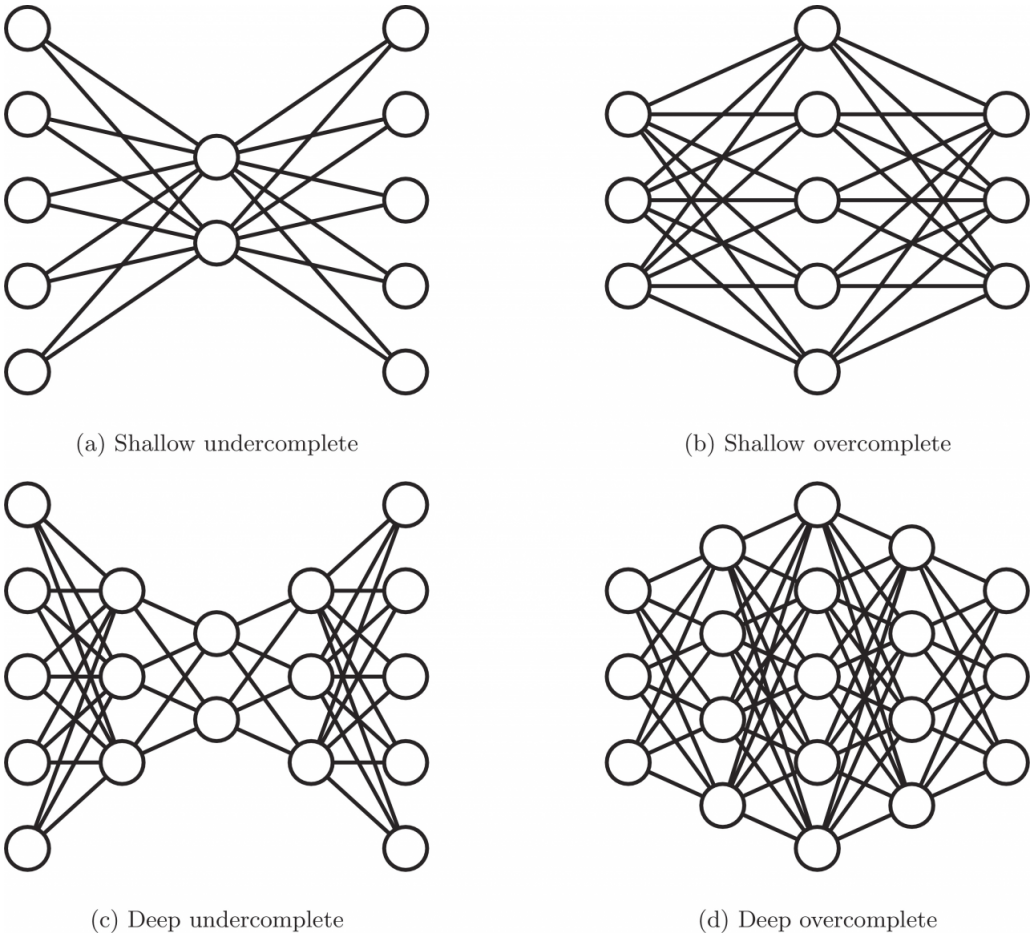


Figure 4.6: Autoencoder example [7]

The taxonomy of autoencoders can vary depending on the application. Two main type of AEs exist based on the dimensionality of the latent representation:

- Undercomplete: In this case, the latent representation dimensionality is lower compared to the input, meaning that the model ends up learning how to compress the data while keeping the meaningful information from it.
- Overcomplete: In here, a higher dimensionality in the latent representation is defined and particular restrictions must be put in place in order to ensure that the model does not just replicate the input onto the output. These type of AEs can be used for Denoising AE applications for instance.

When implementing AEs, non-linear activation functions are commonly defined in the hidden layers. Sigmoid functions and special types of ReLU functions (that consider the negative range of inputs) can be used for this purpose. Also, the AE architecture can be defined with more or less hidden layers (deep or shallow) depending on the application complexity and task [39, 10]. Figure 4.7 illustrate the common taxonomy types of AEs.



(a) Shallow undercomplete

(b) Shallow overcomplete

(c) Deep undercomplete

(d) Deep overcomplete

Figure 4.7: Autoencoder Taxonomy Types [10]

Chapter 5

State of the Art

SEMG-based sensor localization is a relatively poorly studied area; however, important research has been done in the field of biometrics, where EMG, as well as other bio-electrical signals such as electrocardiogram (ECG) [1] and electroencephalography (EEG) [19] have been used for liveness detection, continuous biometric identification and subject verification tasks. Concerning sEMG data processing, machine and deep learning techniques have been widely used recently to extract relevant features from the acquired signals for tasks such as hand-gesture recognition and wrist kinematic estimation. Some relevant work on this field is further presented:

1. Paper 1: *EMG-based online classification of gestures with recurrent neural networks* [41].

   In this study, Feed-Forward Neural Network (FFNN), Recurrent Neural Network (RNN), Long Short-Term Memory network (LSTM) and Gated Recurrent Unit (GRU) are compared for the task of hand gesture classification from sEMG data obtained from UC2018 DualMyo and NinaPro DB5 datasets. The different hand gestures performed by the subjects are shown in Figure 5.1.

Figure 5.1: Dataset hand gestures: (G0) rest, (G1) closed fist, (G2) open hand, (G3) wave in, (G4) wave out, (G5) double tap, (G6) hand down and (G7) hand up [41]

   These two datasets provide a series of activation bursts that are classified in the above mentioned hand gestures, which are ready to be used for supervised training in machine learning and deep learning models.

The results obtained in this study indicate that FFNN, LSTM, RNN and GRU achieved similar accuracy for the previously mentioned datasets (around 95% for DualMyo and 91% for Ninapro). However, LSTM and GRU models used only a third of the parameters, compared to the other models, meaning smaller training and prediction times.

2. Paper 2: *A CNN-LSTM Hybrid Model for Wrist Kinematics Estimation Using Surface Electromyography* [8].

   In this paper, a hybrid model, composed of Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) is proposed for EMG wrist kinematic estimation. For this purpose, sEMG data is collected from six healthy participants, where four wrist movements where performed (flexion-extension, pronation-supination, radial-ulnar deviation and a mix of all of them).

   For the purpose of obtaining the kinematic estimation, a regression output model is built by means of integrating the capabilities of effective feature extraction provided by CNN, along with sequence regression by LSTM. Figure 5.2 displays the block diagram of the hybrid model.

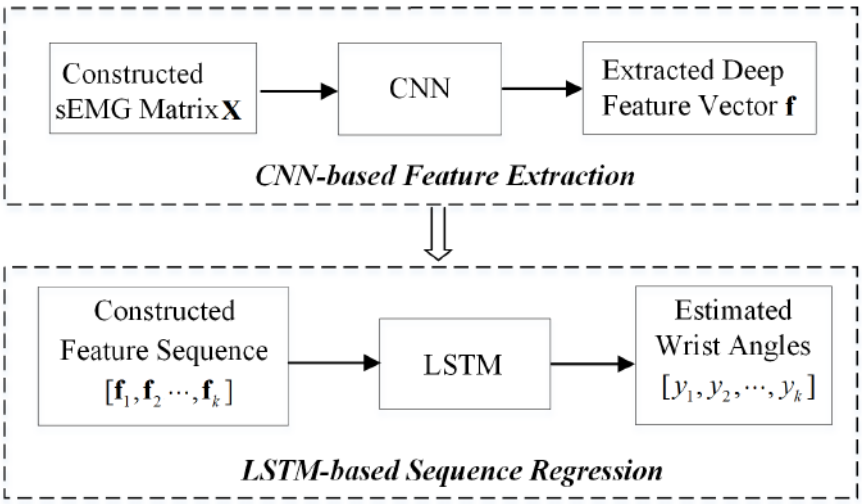


Figure 5.2: Block diagram of CNN-LSTM model [8]

   The hybrid model is compared to CNN and LSTM alone, as well as with ML-based regression models such as Support Vector Regression (SVR) and Random Forest (RF). In this sense, the CNN-LSTM's regressed output is smoother and considerably more similar to the actual ground truth than CNN alone, while also outperforming the previously mentioned models for the flexion-extension, pronation-supination as well as radial-ulnar deviation tasks.

3. Paper 3: *EMGHandNet A hybrid CNN and Bi-LSTM architecture for hand activity classification using surface EMG signals* [22].

   The article presents a hybrid deep learning approach for hand activities classification from sEMG signals. As authors mention, EMG signals have two main components: the spatial sparsity given by the position of the electrodes, and the temporal dependency given by the inherent muscle activity developing over time. Given these two components, the study

proposes the use of Convolutional Neural Networks (CNN), that are capable of extracting deep features from sEMG signals, along with Bidirectional Long Short-Term Memory (Bi-LSTM), that extract bidirectional temporal information, for the purpose of correctly classifying different hand activities. Figure 5.3 presents the complete pipeline that the authors followed for this purpose.

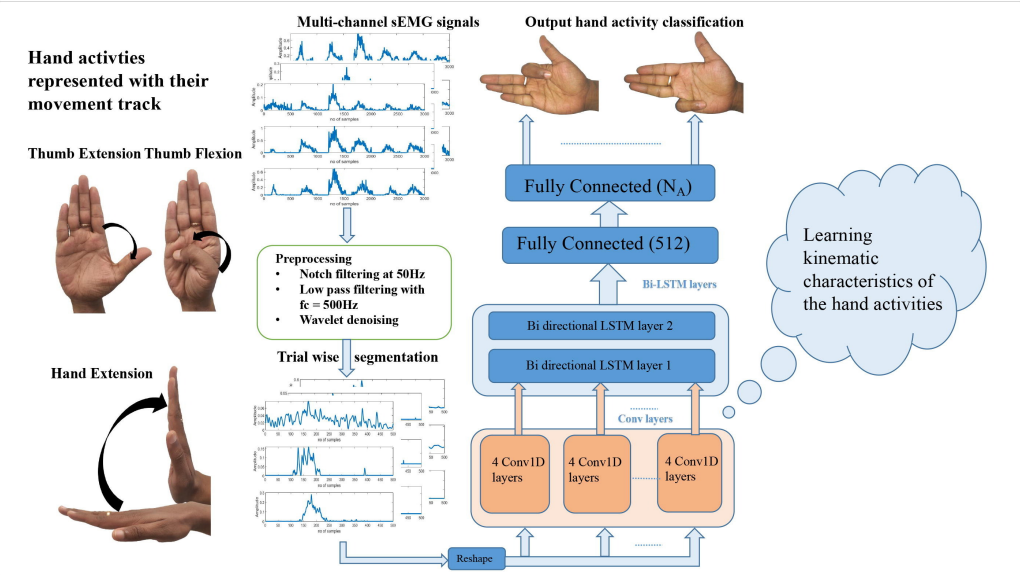


Figure 5.3: Block diagram with the proposed pipeline for the hybrid CNN and Bi-LSTM model [22]

Five different datasets were used for training and testing: NinaPro DB1, NinaPro DB2, NinaOro DB4, BioPatRec DB2 and UCI Gesture. These datasets contain different hand activities/gestures, for which the last dense layer (which contains the softmax activation function) is adapted depending on the number of categories provided by each particular dataset.

The classification accuracy reached by the hybrid model on the previous datasets is 97.77%, 95.9%, 91.65% and 98.33% respectively, outperforming the state of the art models (from 4% to 18.65%)

4. Paper 4: *Hand Gesture Recognition based on sEMG using CNN with Transfer Learning* [11].

   This paper proposes the use of a Transfer Learning technique for sEMG hand gesture recognition. In this vein, a general CNN (the source) is first trained as a feature extractor with a vast dataset containing 30 different hand gestures. Then, transfer learning and non-transfer learning strategies are evaluated in two target architectures separately: CNN and CNN-LSTM (both with the same CNN architecture as the general well-trained CNN). These two target architectures are trained and tested on three different gesture datasets. Figure 5.4 displays the architecture of the source CNN (named ConvNet) and the targets: the CNN-only and the CNN-LSTM with the transfer learning strategy proposed by the authors.

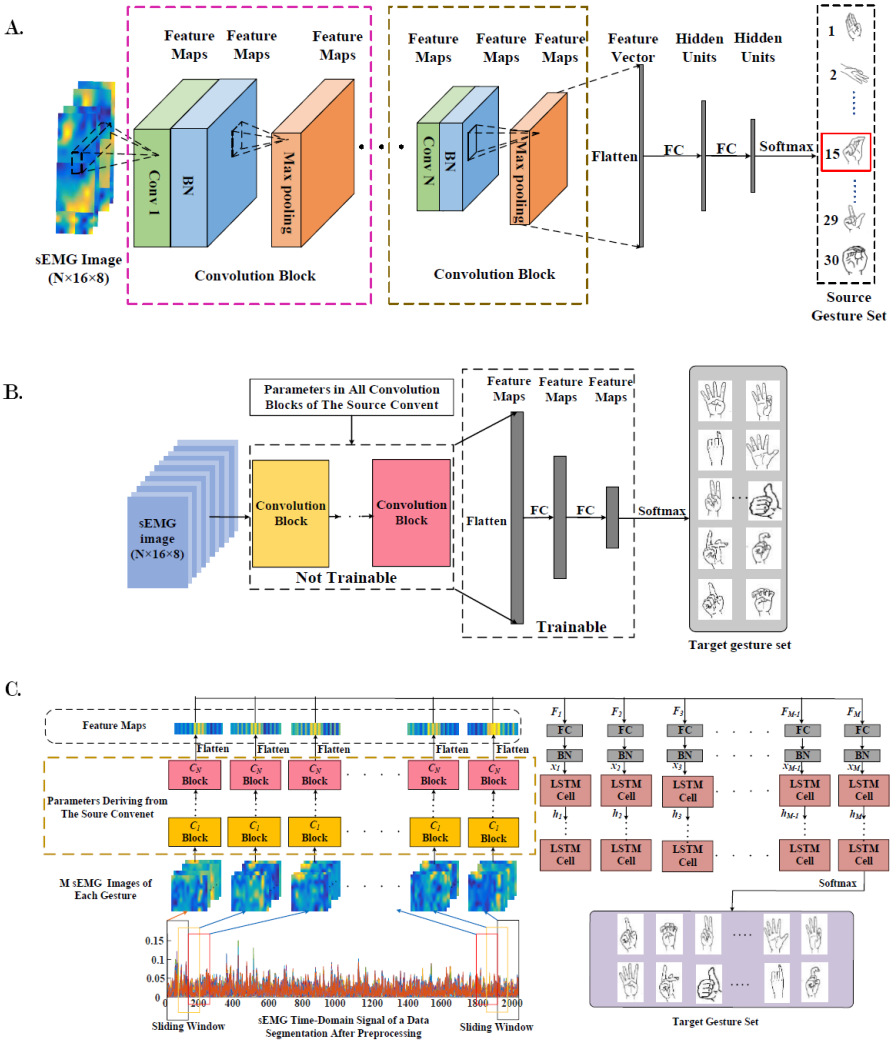


Figure 5.4: A. Architecture of source CNN model. B. Architecture of the target CNN-only model with transfer learning strategy. C. Architecture of the target CNN-LSTM model with transfer learning strategy [11]

> The results from the transfer learning strategy applied to the CNN-only and CNN-LSTM targets revealed improved hand gesture recognition accuracy (10% and 38% respectively) and reduced training time (10 times less) compared to non-transfer learning experiments performed in the study. Recognition accuracy while using the transfer learning strategy reaches 90% with only 2 repetitions per gesture, meaning that simple fine-tuning allows to obtain a substantial accuracy and generalization.

As seen from the state-of-the-art, hybrid models that incorporate CNN and LSTM are demonstrating promising results in tasks like sEMG gesture recognition and kinematic estimation. In this sense, a hybrid model architecture is proposed for the present thesis for the goal of correctly classifying muscles from sEMG data.

Chapter 6

Methods

This thesis proposes the use of surface electromyography to record electrical activity over different muscles of interest with the goal of capturing a series of contraction bursts, from where a hybrid deep learning model is trained with the purpose of inferring on which muscle the sEMG sensor is located. To achieve this goal, the pipeline in Figure 6.1 is proposed.

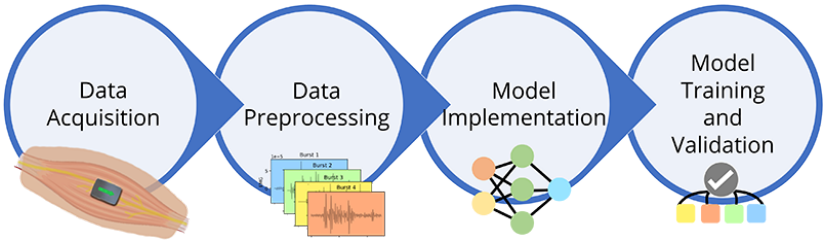


Figure 6.1: Pipeline: Muscle Classification Via Hybrid CNN-LSTM Model

6.1 Data Acquisition

Surface electromyography recordings of contraction bursts from different muscles are required to train and validate the model. For this purpose, Trigno Maize Sensor from the company Delsys is proposed and employed for this study. Note that Delsys is one of the market leaders globally in terms of design and production of a variety of high performance electromyography instruments. Their EMG sensors tackle the main challenges of this technology, such as noise signal artifact, contaminated signal from other muscles (crosstalk), and signal reliability and consistency. Moreover, their EMG instruments come with an easy to use software (Trigno Discover) for the data acquisition and storage. The technical specifications of the sEMG sensor as well as the details of the experimental setup (data collection protocol) are described in the following sub-sections.

6.1.1 Trigno Maize Sensor

Trigno Maize is selected as the sEMG instrument in the present study since it is a reusable high-density sEMG (HD-sEMG) that provides 16 channels for multiple muscle point recordings in real time that allows for the collection of contraction bursts. The device communicates via RF protocol to a receiving base station that can be directly connected to a laptop or desktop, allowing for a wireless setup. This device has a patented technology that guarantees high fidelity signals. Table 6.1 displays the main technical specifications of the sEMG sensor, while Figure 6.2 shows the sEMG sensor along with the charging station and the software.

Table 6.1: Technical Specs of Delsys Trigno Maize sEMG Sensor

Specification	Description/Value
Operating Range	20 m (RF)
Operation Time (Fully charged)	1 to 2 hours
Range (voltage)	11mV
EMG Signal Bandwidth	20 - 450 Hz
Sampling Rate	1000 Hz (Bluetooth)
Number of Channels	16
EMG Contact Dimensions	2 mm
Common Electrode Contact Dimensions	5 x 1 mm
EMG Contact Grid Spacing	6 mm
Contact Material	99.9% Silver
EMG Channel Resolution Depth	17 bits

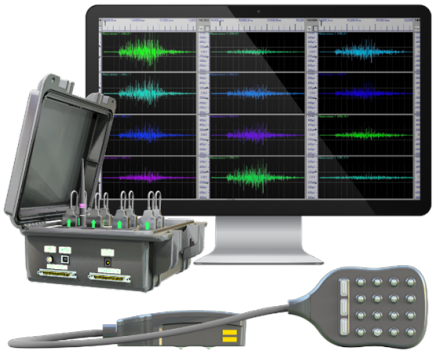


Figure 6.2: Delsys Trigno Maize Sensor, Base Station and Trigno Discover Software [13]

The electrode contact configuration, as well as the channel assignment is presented in Figure 6.3. Note that the Maize sensor use differential electrodes for optimal signal detection. Different active reference contacts improve the technology stability and reduce the captured noise in the signal.

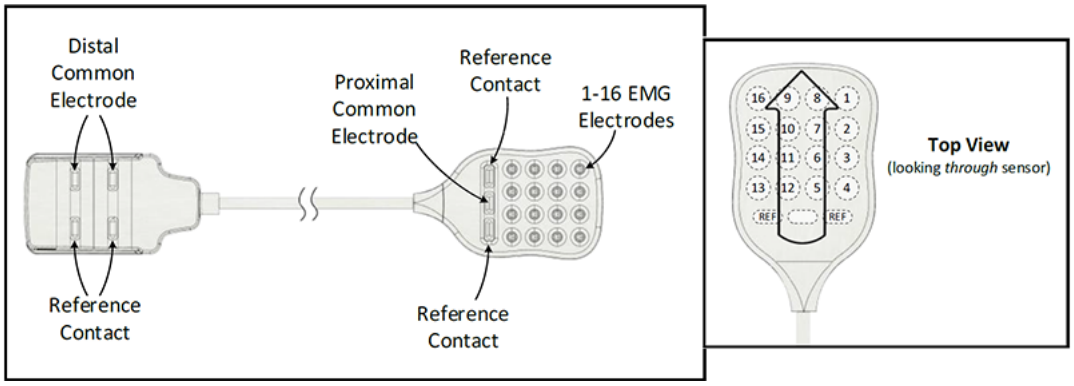
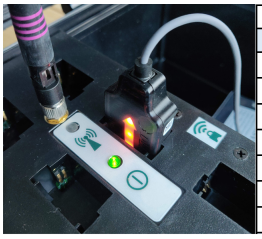


Figure 6.3: Delsys Trigno Maize: Contact Configuration and Channel Assignments [13]

6.1.2 Protocol and Experimental Setup

Nine healthy volunteers participated in the study in Belgium: five men and four women ranging from 25 to 71 years old. None of the participants had any clinical history of neuromuscular disorders that could potentially affect the study. The steps proposed in the protocol as well as the possible risks related to the tests were given to the volunteers. The participants were allowed to stop/leave the session in case of any discomfort. Four muscles were considered for the tests: left and right biceps, as well as left and right tibialis anterior, whereby each recording involved a series of muscle contractions for each of these muscles. The protocol that is used to collect the data is explained step by step below:

1. Ensure the sEMG sensor is correctly charged. The charging station can be used for this purpose. Take into account the color of the sensors to know if they are charging and whether the charge is full. Figure 6.4 displays the sensor while in charging mode. The different power states of the sensor with the corresponding color are presented in the adjacent table (taken from Delsys Maize EMG Sensor User's Guide [14]).



	State	Color	Pattern	Arrow Display
	Common States			
1	Power Off	Off	none	
2	Power On/Activate	White/Green	fade	/
3	Charging	Amber	solid	
4	Charge Complete	Green	solid	
5	Identification Mode	White	rapid flash	/ /
6	Scan (Startup)	Amber/Cyan	slow flash	/
7	Power Up Error	Red	slow flash	/

Figure 6.4: Delsys Trigno Power Modes [14]

2. Prepare and clean the skin surface:

   It's recommended that the area where the sensor is going to be placed does not have excessive hair so that the connection between the contacts and skin is optimal. Moreover, the contact area must be rubbed with a piece of cotton impregnated with isopropyl alcohol

to ensure the removal of oils and residues in the skin. Wait for the skin to dry before continuing.

There are 4 muscles from which sEMG recordings are performed: left and right biceps, as well as left and right tibialis anterior. See muscles in RED in Figure 6.5 as reference. During each test, one muscle at a time is recorded. The proposed sequence is the following: left biceps, right biceps, left tibialis anterior and right tibialis anterior. In this sense, start by cleaning first the left biceps area.

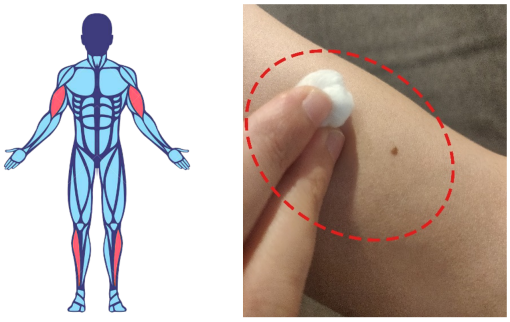


Figure 6.5: Skin Preparation

3. Use a marker to mark on the skin the reference points where the sensor will be located. To detect the EMG activity in the best possible way, the 16 channel sensor must be placed close to the muscle's centroid. Take into account Figure 6.6 to position the mark based on the indicated anatomical landmarks and proportional distances.

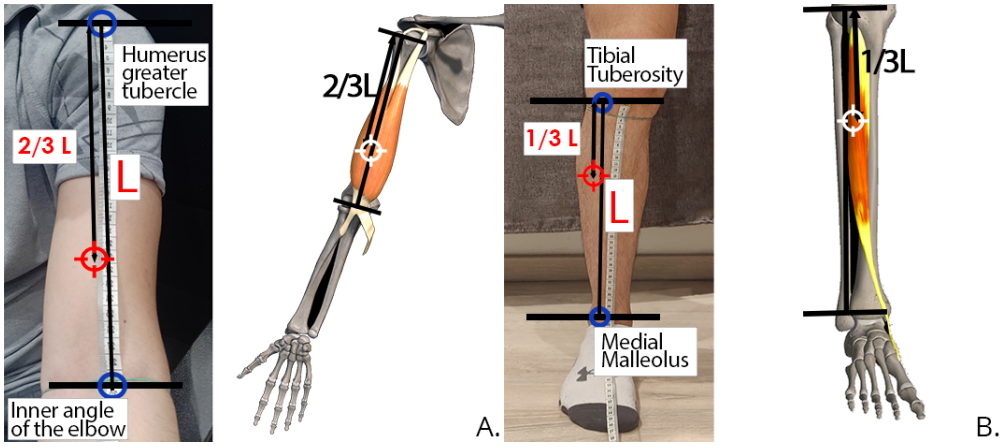


Figure 6.6: Muscle's Centroid Marking (A. Biceps, B. Tibialis Anterior)

4. Run the Trigno Discover Acquisition software on the computer and approach the sEMG sensor to the magnet unlocking system present in the base station. If the sensor starts alternating between yellow and blue color it means it is correctly in SCAN MODE. Then, in the software click Add New Sensor (or press Scan in case it has already been added previously). Use Figure 6.7 as reference.

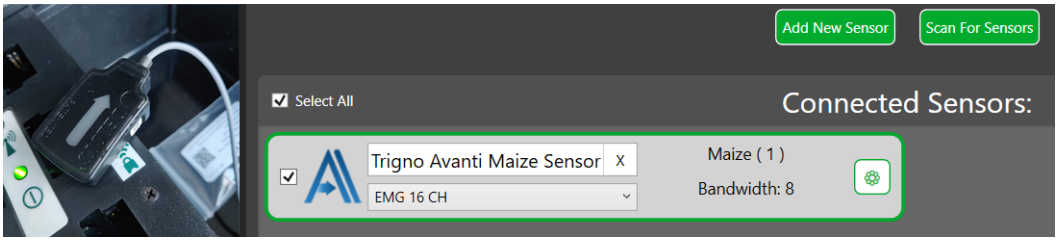


Figure 6.7: Pairing the Sensor to the Base Station

5. Position the 16 channel sEMG sensor on the marked point (starting with the left biceps) and make sure it is well fixed to the surface of the skin. Once the grid sEMG sensor is attached, proceed positioning distal contacts far from the 16 channel electrodes as shown in Figure 6.8. Make use of the adhesive bands provided by the sensor manufacturer (Delsys) to firmly attach the 16 channel sEMG as well as reference electrodes to the skin.

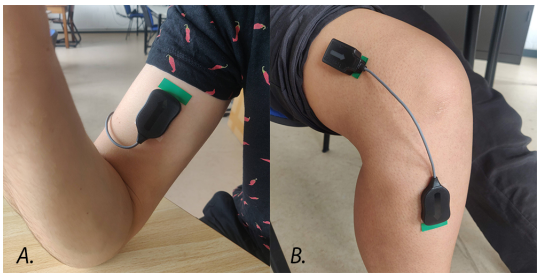


Figure 6.8: Sensor Placement (A. Biceps, B. Tibialis Anterior)

6. Instruct the user on how the muscle contractions must be performed. Figure 6.9 provides a graphical representation of the two main exercises the user needs to perform depending on the muscle that is being currently recorded. For each muscle test, the user must hold the contraction position for around 1 second, then release and go back to relaxed state for around 5 seconds and repeat the process for a total duration of 10 minutes per muscle. Then start the recording by pressing the corresponding button in the software.

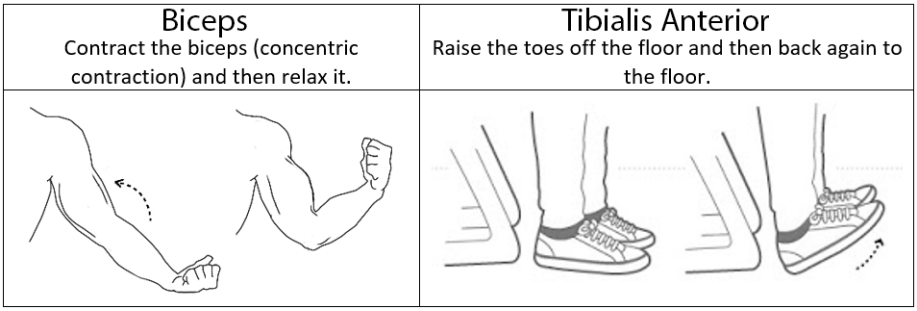


Figure 6.9: Instructed exercises for biceps and tibialis anterior

7. Once the participant completed the proposed time (or decided to quit), proceed to stop the recording and perform a visual check of the overall collected data from the data acquisition software. Check that the 16 channels can be correctly visualized (like seen in Figure 6.10). Subsequently, export the recording in a comma separated value file (csv). Make sure the file is saved in the following location: /subjectno/date/left or right/biceps or tibialisanterior.csv. Follow the naming convention based on the current recorded muscle.

Figure 6.10: Visualized sEMG Recording from Trigno Discover Software

8. Detach the sensors from the participant and repeat the whole protocol for the following muscle, until all four muscles are recorded.

6.2 Data Preprocessing

6.2.1 Collected Data

All subject's sEMG recordings are stored in a comma separated value files. Each recording contains information about the application that recorded the tests (Trigno Discover), the date and time, the collection time length, the id of the sensor and the measured electrical activity across all 16 channels for the duration of the test. Table 6.2 displays a portion of one of the recordings.

Table 6.2: sEMG Collected Data Example

SEMG Measurements (1000Hz)			
Channel 1	Channel 2	Channel 3	Channel 16
-0,00679493	-0,00513	-0,0149	-0,00536
-0,01001358	-0,01025	-0,01979	-0,01144
...	...	...	...
0,01680851	0,018954	0,018001	0,023484

Application	Date/Time	Collection Length	Trigno Avanti Maize Sensor ID
Trigno Discover	15/12/2021 21:58:11	614,844	62366

The summary about the number of collected of burst, the average length duration per subject/muscle and the total average length across all subjects are displayed in Table 6.3. Also, histograms showing the contraction length distribution across the subjects for each of the muscles is provided in Figure 6.11.

Table 6.3: Activation Burst Lengths

Average Burst Lengths per Subject (ms)				
Subject N	Length (Left Biceps)	Length (Right Biceps)	Length (Left Tibialis Ant.)	Length (Right Tibialis Ant.)
1	852.72	925.04	879.43	1029.42
2	1303.35	1275.73	1282.77	1126.35
3	1062.97	1210.15	1197.70	1160.25
4	937.78	1120.66	1428.20	1167.28
5	2296.78	1484.90	1908.92	1921.81
6	675.29	776.44	1336.65	1229.86
7	937.17	963.47	1143.67	1150.25
8	1300.94	1053.05	983.74	891.52
9	1109.46	1289.70	1211.73	1215.80
Tot. Average Length	1164.05	1122.13	1263.65	1210.28
Number of Bursts	1078	1037	1120	1126

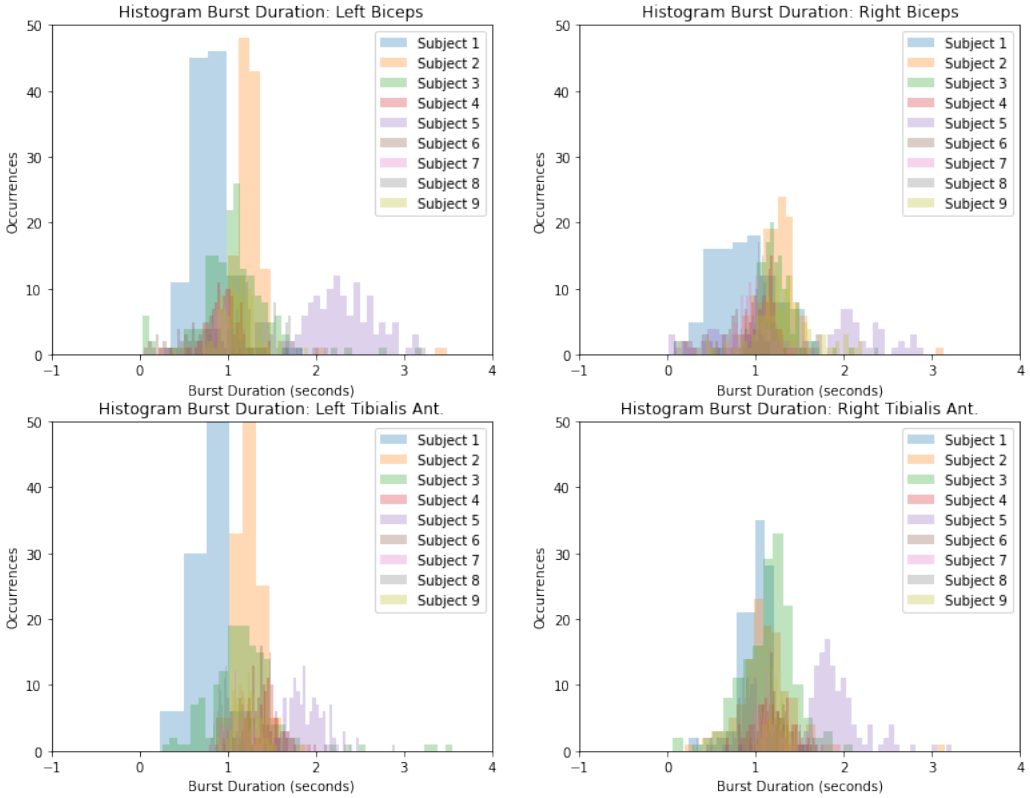


Figure 6.11: Histograms: Activation Burst Lengths

In order to preprocess the data, Python programming language is used to extract the collected data from the csv files and to further do the data processing and model implementation. Figure 6.12 shows 14 seconds of one of the recordings from one subject for a particular muscle.

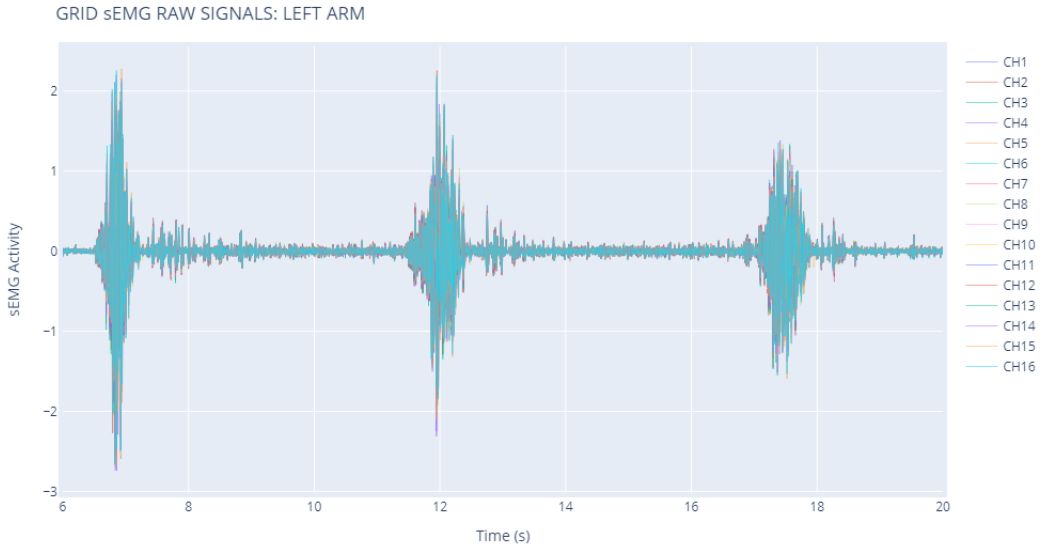


Figure 6.12: SEMG Signal, 16 captured channels

The captured sEMG signals have already a prefiltering provided by the acquisition system: a high pass filter (2-pole Butterworth) and low pass filter (8-pole Butterworth). One can evidence that the behavior of the 16 channels is similar. This is due to the fact that electrodes in the two-dimensional grid configuration are not far from each other (6mm). A zoomed view is provided in Figure 6.13:

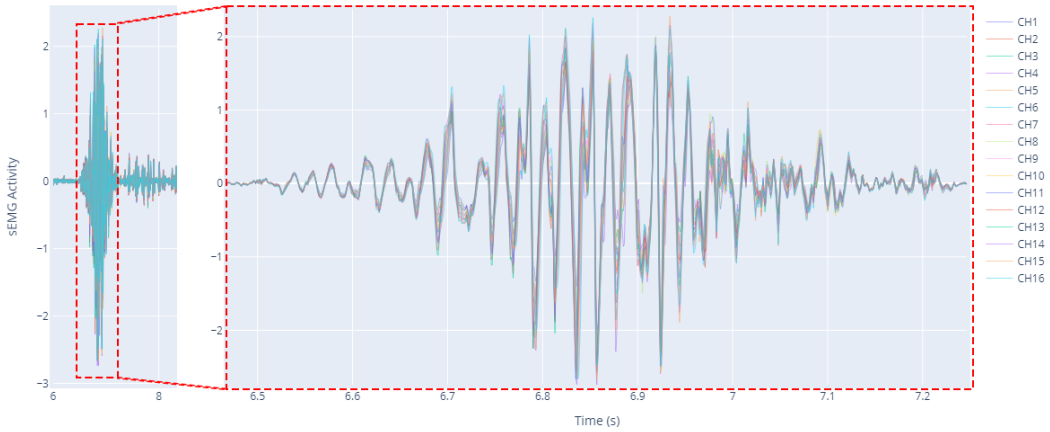


Figure 6.13: Zoom view of recorded activity across channels

6.2.2 Activation burst detection

Throughout the duration of each of the sEMG recordings, two main behaviors can be identified in the signal: the rest period, in which the muscle is in a relaxed state and the electrical activity does not deviate too much from the baseline; and the contraction burst period, where few or multiple motor units activate different muscle fibers, causing an increment in the signal amplitude and frequency. Figure 6.14 shows both behaviors in a portion of the signal.

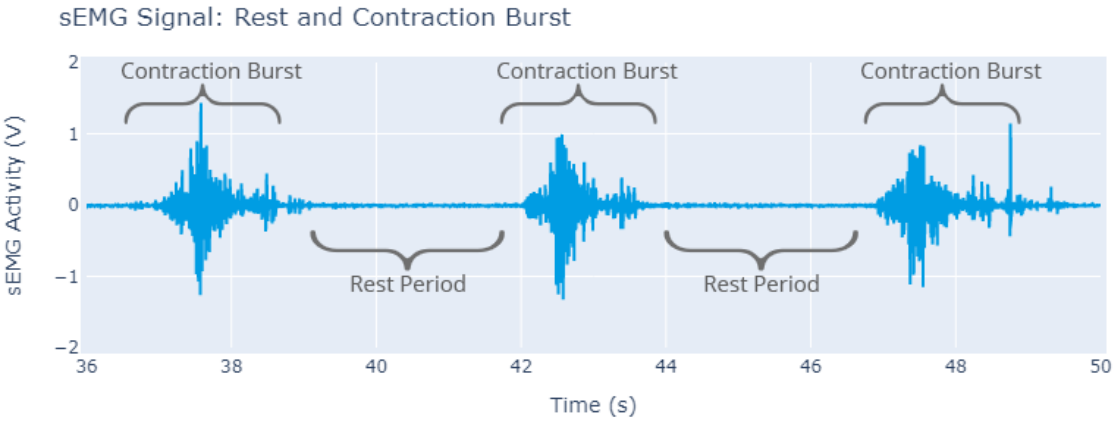


Figure 6.14: Rest period and Contraction Burst

In order to identify the contraction bursts in the sEMG recordings, different type of algorithms can be employed. In this case, an openly available Python library (named biosignalnotebooks [35]) is used for the detection purpose. The library offers a detection function for EMG, in which the beginning and end of each phase of each muscle activation is determined by a single threshold algorithm that makes use of the Teager Kaiser Energy Operator (TKEO), which suppresses the noise and makes the signal larger; thus, making the bursts more easily identifiable. The steps taken by the algorithm in order to detect the activation bursts are the following: (i) a baseline removal is performed where the average electrical activity of the recording is subtracted throughout the whole signal, (ii) the signal is filtered using a Butterworth band-pass filter (selecting the sampling rate of 1000 Hz and low and high cutoff frequencies of 10 and 300 Hz respectively). Posteriorly the TKEO is applied, which basically consist of the following formula:

$$EMG_{(TKEO(t))} = EMG_t^2 - EMG_{(t-1)}EMG_{(t+1)}$$

The signal at the current timestep is squared and the product of the signal of the previous timestep and the following one is subtracted from it. Finally, a moving average is applied to further smooth the signal and a threshold value is used to detect the beginning and end of the activation burst. Further details about the algorithm implementation are more deeply explained in biosignalsnotebooks repository and project page [34]. Figure 6.15 shows the detected activation signals obtained from the proposed algorithm.

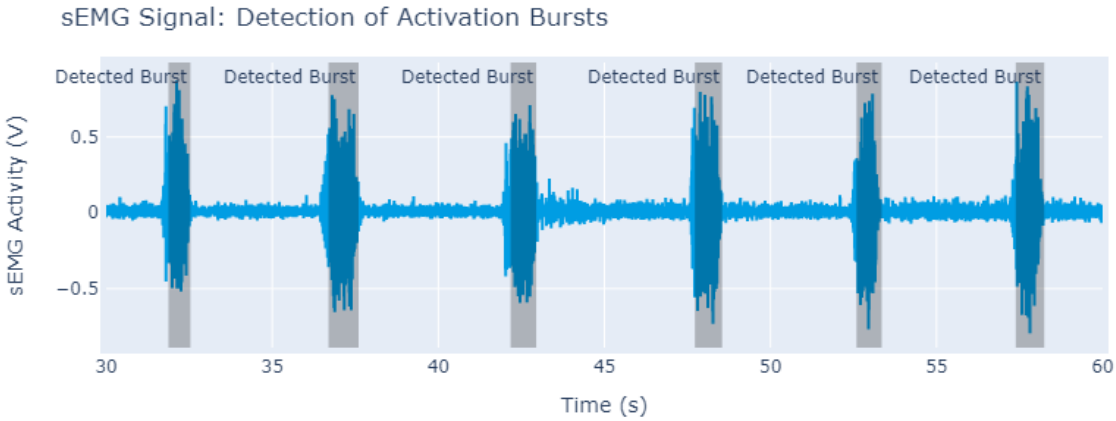


Figure 6.15: Contraction Burst Detection

6.2.3 Onset correction and window sized inputs

Since contraction bursts duration can vary substantially between subjects and even for the same subject throughout the test (see Figure 6.11), only 500 milliseconds of the signal are considered (prioritizing the onset of the contraction). Therefore, a small correction to the previous algorithm is applied to cover more time-steps (100 milliseconds) before the original onset as detected by the algorithm. Figure 6.16 displays the detected start and end of contraction burst by the algorithm for a particular burst (region in gray) and the corrected onset (region between red dotted lines). From the start of the corrected onset, a window of 500 milliseconds is defined. This fixed sized window is also set as the sample's input size for the deep learning model that is discussed in a posterior section.

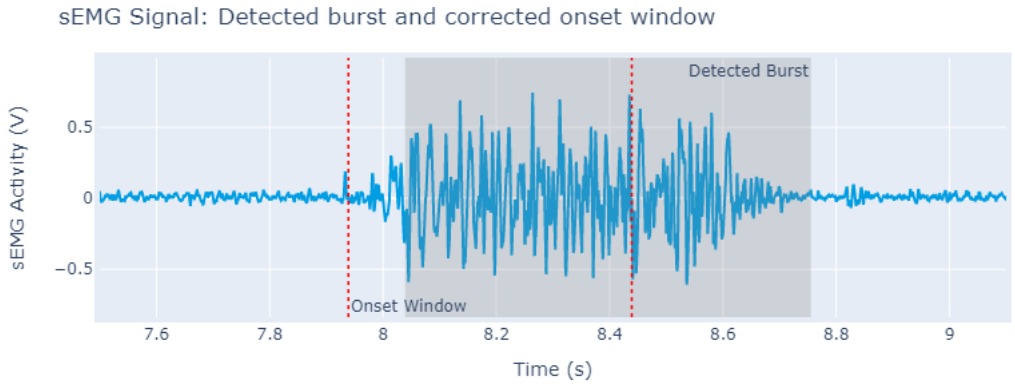


Figure 6.16: Detected burst and corrected onset window

6.2.4 Data Augmentation

The 16 channels offered by the sEMG sensor in the two-dimensional grid configuration are relatively close to each other (6 mm). Thus, the electrical activity in an activation burst does not deviate dramatically between these channels but rather follows a similar pattern as shown in Figure 6.13. In this sense, the 16 signals identified in each burst were used independently (as separate samples), as a sort of data augmentation strategy. It is important to highlight that even though the signals on each of the channels were treated as separate samples for each burst, the signals of the channels for a particular burst in the training set and validation set were not mixed since these channel signals are highly correlated for the same burst.

From the total 4361 contraction bursts obtained from the experiments with the participants, a total of 69776 data samples are obtained after splitting bursts into the 16 channels. These samples are used for the training and validation of the hybrid deep learning model described in a following section.

6.3 Model Implementation

As previously reviewed from the state-of-the-art in Chapter 5, sEMG signal processing is nowadays being approached by hybrid deep learning techniques with better results compared to more traditional machine learning techniques. The models with higher performance and accuracy normally involved Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks for tasks such as gesture recognition and wrist kinematics estimation. Based on this, a hybrid CNN-LSTM is proposed in the current thesis as the architecture to address the muscle classification task. This hybrid strategy allows the model to exploit the spatial and temporal information that is inherent in sEMG recordings.

6.3.1 Dataset Distribution

The dataset distribution for the training and validation of the model contemplates two validation scenarios that allows to assess the intrapersonal and interpersonal accuracy of the model. Figure 6.17 illustrates how the dataset is distributed.

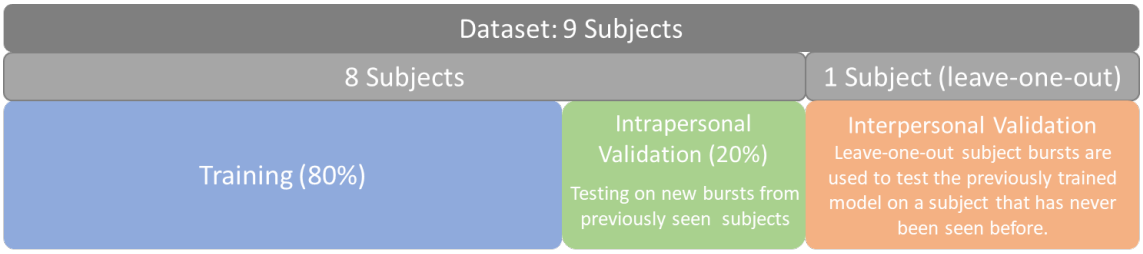


Figure 6.17: Dataset Distribution: Training and Validation

A Intrapersonal Scenario Evaluation:

In this scenario, the idea is to assess the ability of the model to correctly classify new contraction bursts from previously seen participants in the training step, meaning that all subjects are involved in the training and validation step except for the leave-one-out subject.

B Interpersonal Scenario Evaluation (Leave-one-out):

In this second scenario, the model is tested on its ability to correctly classify new contraction bursts from unseen participants in the training step. Meaning that a subject is completely excluded from the training and it's only used to validate the model. The validation in this scenario is composed by all contraction bursts from the excluded subject.

6.3.2 Model Architecture

Inspired by state-of-the-art models [8, 22, 41, 11] discussed in Chapter 5, a hybrid model architecture is proposed for the task of multiclass classification, more concretely: muscle classification from sEMG activation burst signals. The model is composed of CNN, LSTM and FC layers, taking advantage of CNN layers as feature extractors and the LSTM layer to maintain long-term dependencies that are inherent in sEMG signals. Figure 6.18 represents the architecture employed, while Table 6.4 describes in more detail the hyperparameters and dimensions in each of the layers of the architecure. The input tensor goes through two CNN layers followed by a FC layer, similar to one of the architectures presented by Xiang Chen et al.[11], which then goes to one LSTM layer. Notice that LSTM layers are prone to overfit more easily than CNN layers [8], and for this reason only one layer of LSTM is defined for this study. Finally the output of the LSTM goes through a second FC layer and then a last FC with a softmax function providing the output tensor, which corresponds to the probability of the four muscle classes. Take into account that some of the hyperparameters are determined via empirical manual tuning.

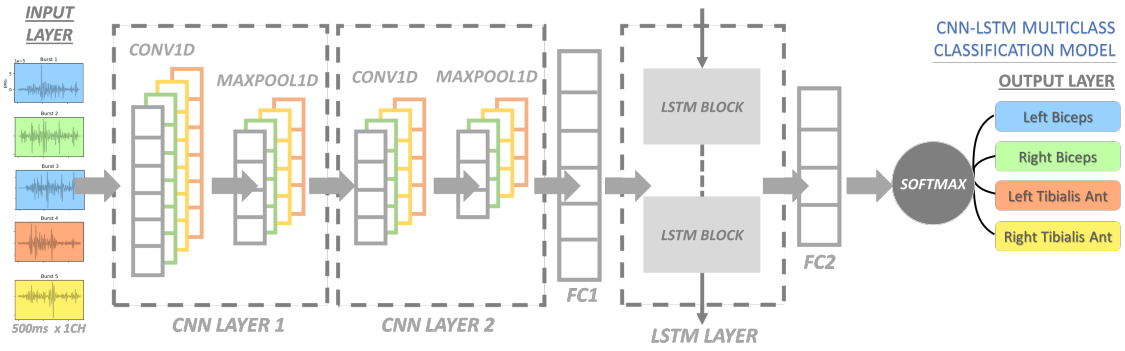


Figure 6.18: Hybrid CNN-LSTM Model Architecture

Table 6.4: Hybrid CNN-LSTM Model Architecture: Layer Hyperparameters

Model Architecture: Hybrid CNN-LSTM for Muscle Classification			
Layer Type	Definition	Output Shape	Parameters
Input Layer	sEMG Contraction Burst	500x1	0
CNN1 Conv1D	N.Filters = 32 Kernel Size = 5x1 Stride = 1 Activation: LeakyReLU (Slope = 0.1)	500x32	192
CNN1 MaxPool1D	Kernel Size = 2x1 Stride = 2	250x32	0
Dropout	30%	250x32	0
CNN2 Conv1D	N.Filters = 64 Kernel Size = 7x1 Stride = 1 Activation: LeakyReLU (Slope = 0.1)	250x64	14400
CNN2 MaxPool1D	Kernel Size = 2x1 Stride = 2	125x64	0
Dropout	30%	125x64	0
Fully Connected 1	50 units Activation: LeakyReLU (Slope = 0.1)	125x50	3250
Dropout	30%	125x50	0
LSTM Layer	LSTM Blocks = 30	30	9720
Dropout	30%	30	0
Fully Connected 2	30 units Activation: LeakyReLU (Slope = 0.1)	125x50	465
Dropout	30%	125x50	0
Output Layer	Fully Connected + Softmax	4	64

- Input Layer: Each input sample that is provided to the model has size 500x1, corresponding to the defined window size that is captured in the sEMG detected contraction burst for a particular channel. Notice that samples are randomly shuffled and are provided to the model in batches of 1024 in form of TensorFlow record files (TFRecords).
- CNN Layers: Two layers of Convolutional Neural Networks are proposed to extract relevant features from the contraction bursts of the four captured muscles. In this sense, successive convolutions over the input sEMG array are performed in an iterative process that enables the model to extract pertinent features from the data. This is then followed by a non-linear activation function: Leaky ReLu (which tackles the vanishing gradient problem and introduces a negative slope for negative values). Finally, after the non-linear activation, a Max Pooling operation is applied to reduce the dimensionality by calculating the largest value of a defined kernel size over each of the previously obtained feature maps. Notice that Max Pooling allows to reduce the computational cost and also helps reducing the problem of overfitting.
- LSTM Layer: the Long Short-Term memory layer is used to maintain temporal information in memory for the sequence of data obtained from previous layers. In this sense, feedback loops from previous time-steps influence the output at the current one. The whole working

principle of LSTM is explained more deeply in previous Chapter 4.

- FC Layers: Fully Connected layers are feed forward neural networks in which the inputs of a layer are completely connected to all activation units of the following layer through a weight matrix. These layers receive high-level features in the data (in our case from the second CNN layer and the LSTM layer) and learn non-linear combinations of the received input features.
- Dropout Layers: After each CNN layer, LSTM layer and FC layers, a dropout rate of 30% is applied as a regularization technique, which helps the overall deep neural network to avoid overfitting. This technique randomly selects hidden units and temporarily disables them from updating their parameters during the training phase. This counteracts the effect that the model relies only on particular units, but rather forces all units to learn resulting in a more generalizable model after training.
- Output Layer: One last fully connected layer provides the outputs corresponding to the classes (in the case of this study, 4 muscle classes), followed by a Softmax activation function that provides the probabilities for the four different muscles that are considered in the classification task.

6.3.3 Pretraining: Layer Wise Unsupervised Learning on CNN layers

Due to lack of publicly available pretrained models processing the sEMG signal, it was decided to use autoencoders to obtain optimal initialization weights[1] for part of the designed network. A Greedy layer-by-layer unsupervised pre-training strategy [3] is proposed to obtain the weight initialization of the two CNN layers. This method is known to provide very good generalization properties and to overcome problems of local optimization during the training phase. In this vein, autoencoder architectures are built to implement a reconstruction task for each of the CNN layers, meaning that each of these layers is isolated and used in the encoder part of its own specific autoencoder for recontsruction. An encoder-decoder architecture is thus put in place, where the encoder is driven by the CNN layer, while the decoder is attached in form of a deconvolution operation that allows reconstructing the input tensor from the latent representation. The process is described in more detail in the remainder for this section. A generic explanation on the principle of autoencoders is given in Chapter 4.

Notice that by imposing the reconstruction task upon the autoencoder architecture, the model ends up learning the essential features to produce a latent representation (encoding part) of the data provided in the input tensor, which is reconstructed by the decoder. The loss function used for both architectures is the Mean Squared Error (MSE), with a defined learning rate of 0.001, and using ADAM as optimizer over 100 epochs.

[1] Optimal weights are obtained after training the AE for the reconstruction task and these are used to initialize part of the designed model

- ***First CNN Layer Pretraining:***

  The autoencoder architecture proposed for the first CNN layer is shown in Figure 6.19. In this case, the input tensor corresponds to the sEMG contraction burst signals, which are contained in 500 timestep windows. A one dimensional convolution of 32 filters with kernel size 5x1 and stride 1 is applied to that input window, followed by a non-linear activation (LeakyReLU) and posteriorly a one dimensional pooling operation of kernel size 2x1 and stride 2 to reduce the dimensionality and produce a latent representation. Then, a one dimensional deconvolution with kernel size 5x1 along with the same non-linear activation as before is used to reconstruct the input tensor. The dataset used for this AE training and validation corresponds to the same dataset used for the final model (ignoring the label).

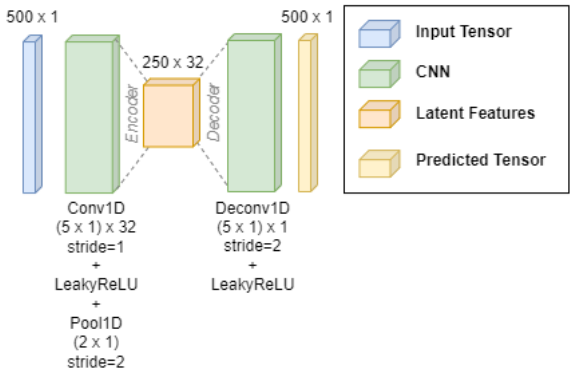


Figure 6.19: Layer-wise unsupervised pretraining for first CNN layer: Reconstruction task

- ***Second CNN Layer Pretraining:***

  Similar to the previous CNN layer, an autoencoder architecture is used for the reconstruction task on this layer as well. The input tensor in this case corresponds to the output of the previous CNN layer (250x32). A one dimensional convolution of 64 filters with kernel size 7x1 and stride 1 is applied to the input tensor, followed by a non-linear activation (LeakyReLU) and posteriorly a one dimensional pooling operation of kernel size 2x1 and stride 2 that produces the latent representation. Finally, a one dimensional deconvolution with kernel size 7x1 along with the same non-linear activation is used for the reconstruction task. Figure 6.20 illustrates the AE architecture for this layer. The dataset used for this AE training and validation is generated by using the encoder outputs from the previous AE.

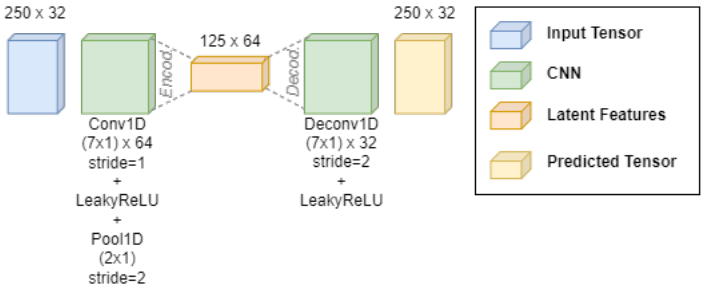


Figure 6.20: Layer-wise unsupervised pretraining for second CNN layer: Reconstruction task

The parameters learned by the encoder part from both CNN layers are posteriorly transferred and used in the final model as shown in Figure 6.21.

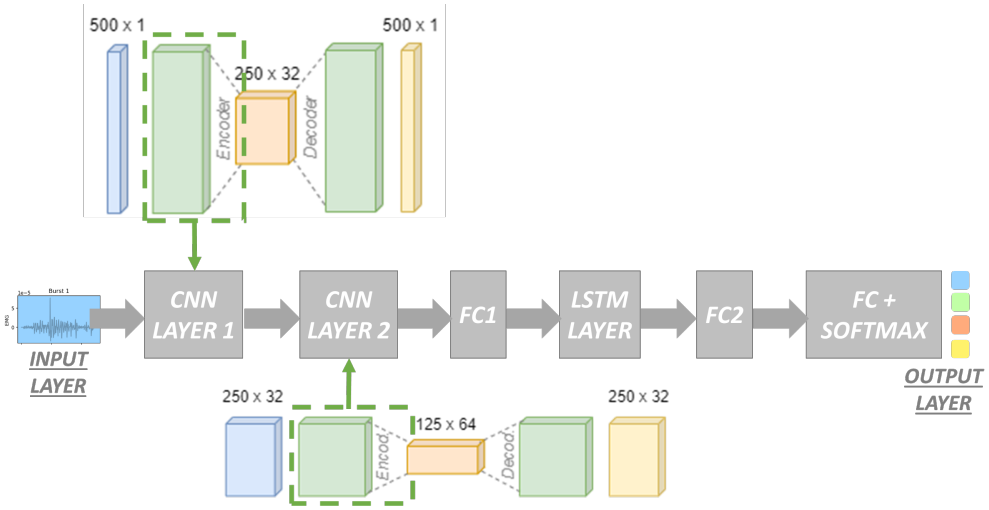


Figure 6.21: Parameter Transfer from Pretraining to the Final Model (Weight Initialization)

6.3.4 Training Setup

The hybrid CNN-LSTM model is trained over 200 epochs, using batches of 1024 samples at a time in form of TensorFlow Records. The selected loss function is the categorical cross entropy, which is an ideal loss function for the multiclass classification task that the model is addressing. Moreover, the accuracy is stored through the epochs as it is a relevant metric in classification. Adaptive Moment Estimation (ADAM) is used as optimizer with a fixed learning rate of 0.001. Note that some of the hyperparameters defined in the architecture of the model are determined through empirical manual tuning. A total of 49648 samples are used for the training step, while 12464 samples are used for the first validation scenario (new bursts from seen subjects) and 7664 samples for the second validation scenario (new bursts from unseen subject, meaning the leave-one-out subject). Model training checkpoints are used through all epochs to store the best

model at the end of the training step, which is later used for validation purposes. Two training setups are defined here: A first setup uses the weight initialization provided from the layer wise unsupervised pretraining step (while keeping CNN layers unfrozen for further parameter fine-tuning) and the second one uses random weight initialization. The latter is included for comparative evaluation to verify how influential this layer-wise pretraining is for the accuracy in the validation scenarios. Model training and validation are run on tensorflow (version: 2.6.2).

6.3.5 Evaluation Metrics

The following metrics are defined to assess the performance of the model for the task of muscle classification (multi-class) on the two proposed validation scenarios after the training step is completed.

- Accuracy: This metric is the percentage of correct predictions made with respect to the total number of predictions. Accuracy expresses how often predictions match the labels.

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FalseNegative}$$

- Precision: This metric is the ratio of correct true positives over the total number of predicted true positives (including true positives and false positives). Precision expresses how reliable the model classifies positives.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

- Recall: This metric is the ratio of correct true positives over the total number of predicted positives (including true positives and false negatives). Recall expresses the model's ability to detect positives.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

- F1-Score: This metric takes into account the precision and recall by following the formula:

$$F1_{Score} = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

  Notice that for multiclass classification the F-1 Score is calculated per class in one-vs-rest approach.

- Confusion Matrix: Summarizes the performance of the classification model in a matrix of predictions vs label for each of the classes (4x4 matrix in this case).

Chapter 7

Results

7.1 Pretraining Results

The results obtained from the pretraining of the two CNN layers are shown in Figure 7.1 and Figure 7.2 respectively. This step is used to obtain better generalization and overcome local optimization problems during the final training phase. In this sense, optimal weight initialization for the two CNN layers are obtained after the pretraining step and these are provided to the final model (see Figure 6.21) in one of the proposed training scenarios for the hybrid CNN-LSTM model. A layer wise unsupervised pretraining is performed using autoencoder architectures for the reconstruction task, which allows each independent CNN layer to learn the essential features from the input tensors to produce a latent representation.

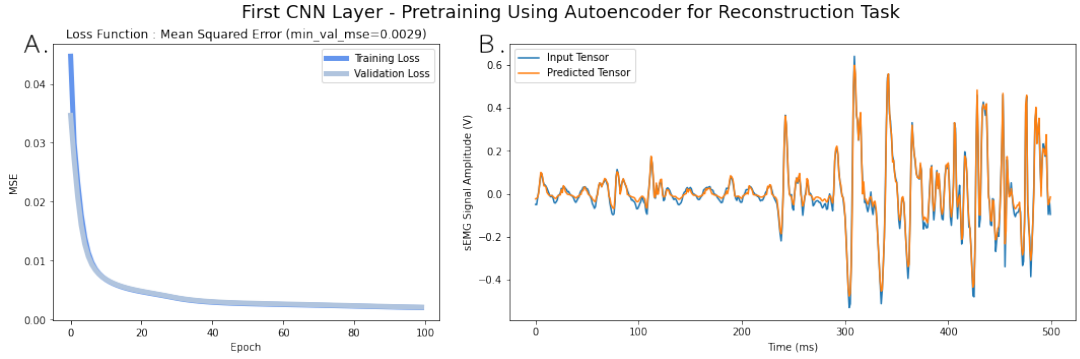


Figure 7.1: Layer-wise unsupervised pretraining for first CNN layer (A. Loss over Epochs, B. Input Tensor vs Predicted Tensor)

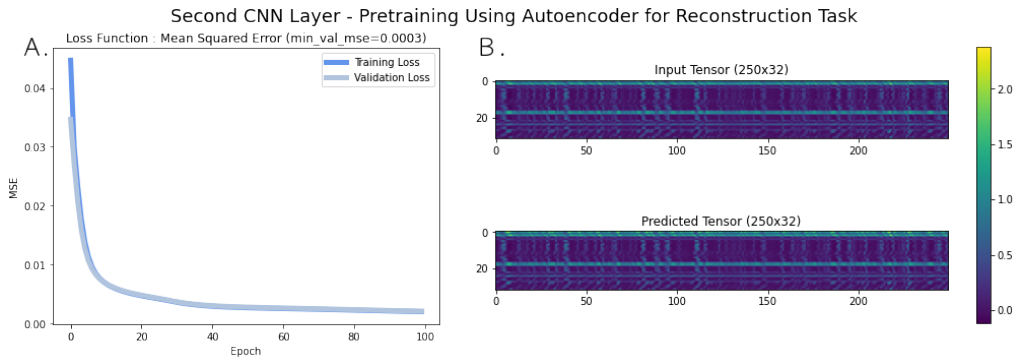


Figure 7.2: Layer-wise unsupervised pretraining for second CNN layer (A. Loss over Epochs, B. Input Tensor vs Predicted Tensor)

The minimum loss value reached in the first AE after completing the training was 0.0029, while the second AE achieved 0.0003. The proposed strategy for the CNN layer wise pretraining is evaluated by training the final model with and without the learned weights from this pretraining step. The results indicate that the model is around 3% more accurate in the intrapersonal validation dataset when using the mentioned pretraining strategy. However, for the interpersonal validation, the same accuracy was achieved with and without weight initialization from pretraining. These results are shown in detail in the following subsection since they are evidenced only after the training and validation of the final hybrid model.

7.2 Training and Validation Results

Two training setups are presented here: A first one using the weight initialization provided from the layer-wise unsupervised pretraining step and the second one that only uses random weight initialization. After training the model for 200 epochs, the model was able to achieve 95.0% (setup 1) and 92.56% (setup 2) classification accuracy in the validation dataset (intrapersonal scenario), where new contraction bursts from already seen subjects where given to the model for the validation purposes. Concerning the second proposed validation scenario (interpersonal scenario or leave-one-out), the model achieved 48% accuracy (in both setups) for muscle classification of contraction bursts from an excluded subject in the training. Something interesting that can be seen from Figure 7.4 is that when there is a misclassification in the interpersonal validation, it is often between symmetrical muscles, meaning for instance that left biceps is more likely to be misclassified as right biceps and viceversa (also for left and right tibialis anterior). This behaviour ends up heavily penalizing the overall accuracy of the model in this validation scenario. The evolution of the loss, as well as the validation accuracy over the epochs is presented in Figure 7.3. The performance metrics obtained for the training (for the two setups) and the two validation scenarios are presented in Table 7.1. Finally, confusion matrices are presented in Figure 7.4.

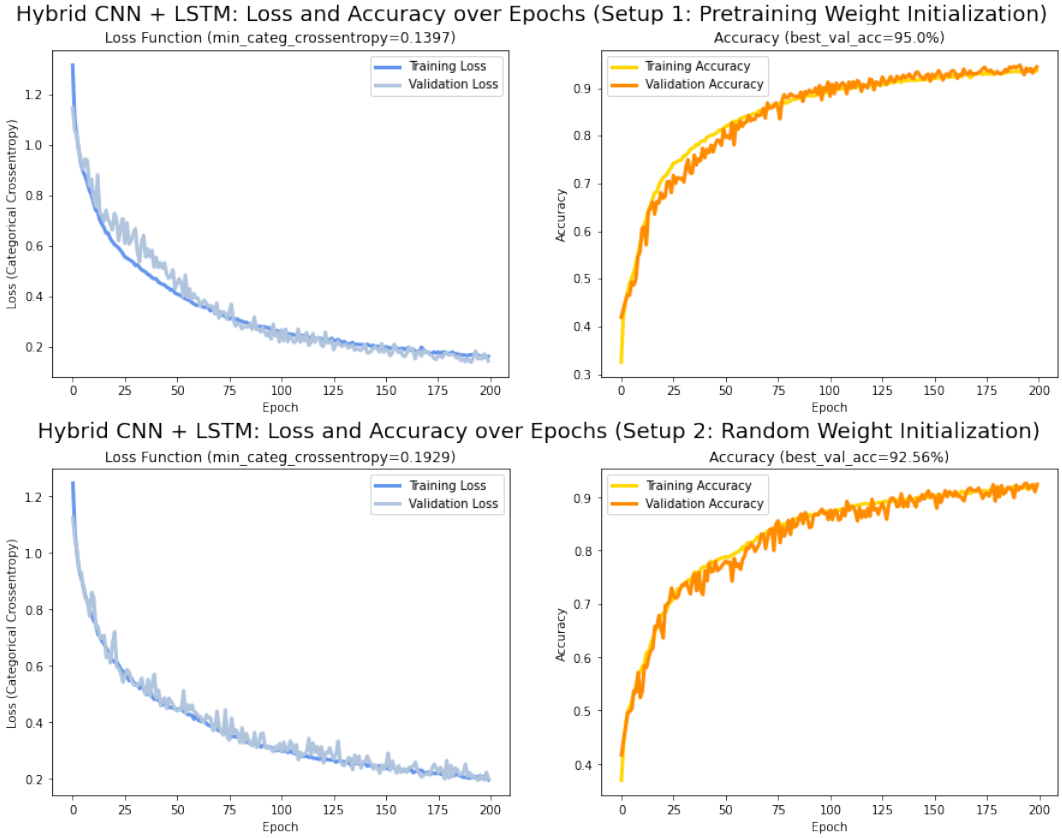


Figure 7.3: Training and Intrapersonal Validation: Loss and Accuracy over Epochs for the two training setups

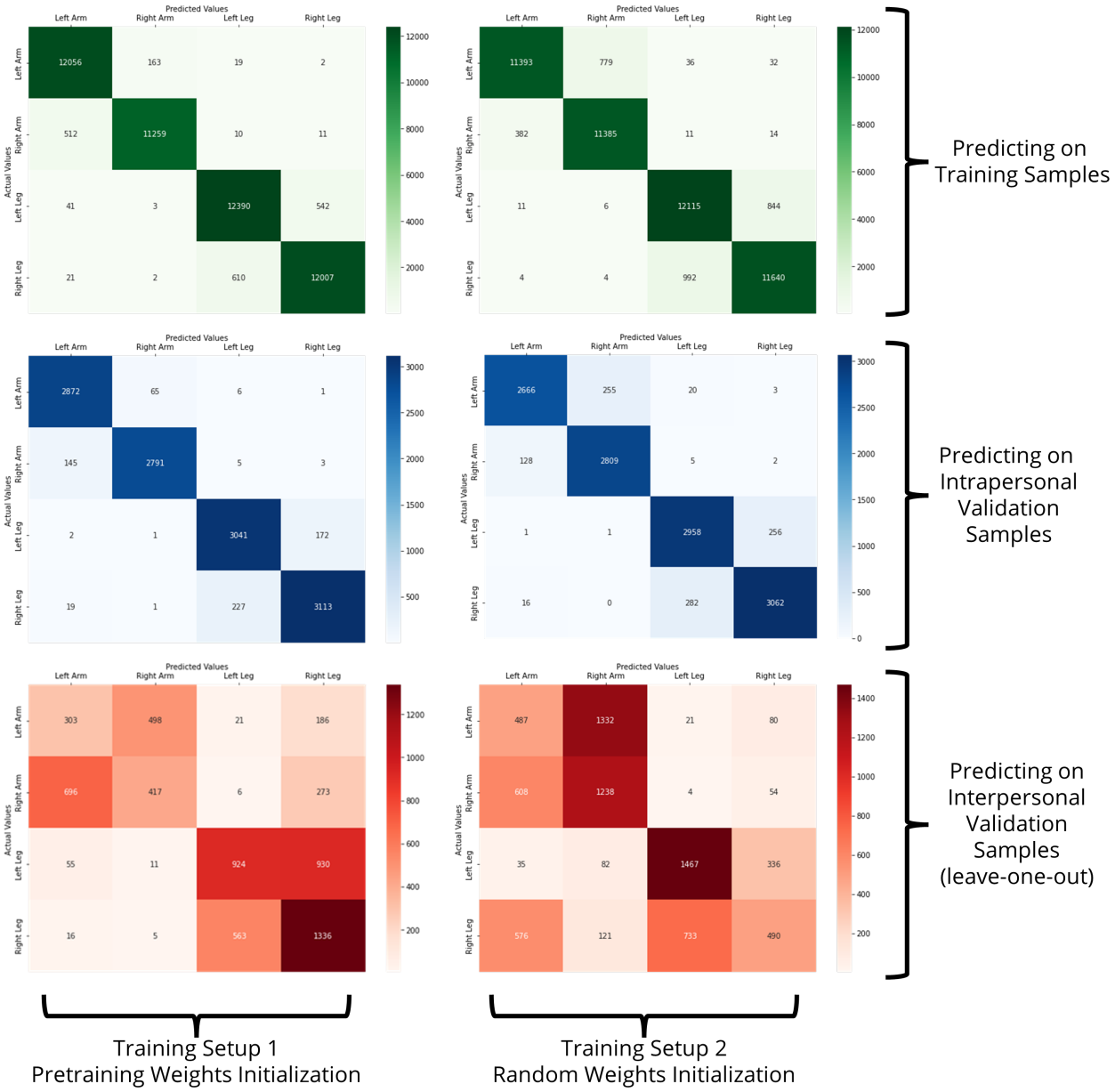


Figure 7.4: Confusion Matrices: Predictions for Training, Intrapersonal Validation and Interpersonal Validation Datasets

Table 7.1: Evaluation Metrics: Predictions for Training, Intrapersonal Validation and Interpersonal Validation Datasets

Training	Left Biceps		Right Biceps		Left Tibialis Ant.		Right Tibialis Ant.	
	Setup 1	Setup 2	Setup 1	Setup 2	Setup 1	Setup 2	Setup 1	Setup 2
Precision	0,955	0,966	0,985	0,935	0,951	0,921	0,956	0,929
Recall	0,985	0,931	0,955	0,965	0,955	0,934	0,95	0,921
F-1 Score	0,97	0,948	0,97	0,95	0,953	0,927	0,953	0,925
	Setup 1				Setup 2			
Accuracy	0,96				0,93			

Intra. Valid.	Left Biceps		Right Biceps		Left Tibialis Ant.		Right Tibialis Ant.	
	Setup 1	Setup 2	Setup 1	Setup 2	Setup 1	Setup 2	Setup 1	Setup 2
Precision	0,945	0,948	0,977	0,916	0,927	0,906	0,946	0,921
Recall	0,976	0,926	0,948	0,954	0,946	0,92	0,926	0,911
F-1 Score	0,96	0,926	0,962	0,935	0,936	0,913	0,936	0,916
	Setup 1				Setup 2			
Accuracy	0,95				0,92			

Inter. Valid	Left Biceps		Right Biceps		Left Tibialis Ant.		Right Tibialis Ant.	
	Setup 1	Setup 2	Setup 1	Setup 2	Setup 1	Setup 2	Setup 1	Setup 2
Precision	0,324	0,285	0,419	0,446	0,666	0,659	0,535	0,51
Recall	0,424	0,254	0,441	0,65	0,931	0,764	0,13	0,255
F-1 Score	0,367	0,269	0,43	0,529	0,777	0,708	0,209	0,34
	Setup 1				Setup 2			
Accuracy	0,48				0,48			

Chapter 8

Discussion

In general, muscle classification from sEMG contraction bursts is a non-trivial task starting by the fact that sEMG signals are prone to a number of factors that can alter the recordings, such as electrical noise, motion artifact, contact between electrodes and skin, sensor positioning with respect to the muscle and contamination from neighbor muscle signals. Moreover, physiological and anatomical factors have as well an influence in the recorded signal: muscle size, applied contraction force, fatigue and the presence of abnormal behavior in the signals. All these factors increase the task complexity, and it is part of the reason why deep learning techniques are being used with this kind of signals. In regards of the architecture defined in the present thesis, a particular CNN-LSTM architecture was trained and tested; however, more variations of this architecture are yet to be explored. For instance, having more than one LSTM layer, using Bi-directional LSTM, increasing the number of CNN layers, and potentially the use of different transfer learning approaches can as well lead to different results. Similarly, different transformations (popular in similar domains, e.g. voice processing) applied to the sEMG input data that involve time-frequency are yet to be investigated in a future study, like Short-time Fourier Transform (STFT), Wavelet Transform (WT), among other.

Concerning the dataset size, contraction bursts from nine participants over four different muscles were collected for training and validation purposes. However, the collection and inclusion of sEMG data from more subjects can potentially lead to a model that can generalize muscle contractions better, improving the results for both validation scenarios.

To get the weight initialization of the two CNN layers, a Greedy layer-by-layer unsupervised pre-training strategy is implemented. This approach is renowned for improving generalization and resolving issues with local optimization during the training phase. In this process, autoencoder architectures are developed for reconstruction tasks for each CNN layer, which entails that each layer is separated and trained accordingly for this task and then the learned weights are transferred to the final model. The use of this strategy provided a slight increase in the intrapersonal validation scenario (3%), but not in the interpersonal validation.

The intrapersonal validation results suggest that the model can be potentially used to classify new contraction bursts from subjects that have already been involved in the training step. At least nine out of ten new contraction bursts from seen subjects are correctly classified by the model (95% accuracy when using pretrained weight initialization and 92% when using random weights). On the other hand, interpersonal classification accuracy did not achieve an acceptable accuracy (only 48% accuracy). In this regard, muscle classification among symmetrical muscles from unseen subjects (leave-one-out scenario) seems a true challenge for the model. This may be partially due to the fact that symmetrical muscles have anatomical and functional similarities which hamper direct person to person generalization with a deep learning model when it comes to their classification. Further testing of this validation scenario with the inclusion of more subjects is required to validate this similarity hypothesis.

A third validation scenario using posterior session recordings from previously seen subjects would be interesting to be considered in a future study so as to assess if the classification accuracy of the model is maintained throughout different sessions. This is specially relevant considering the number of electrical, anatomical and physiological factors that influence the recordings.

Future studies can make use of the collected sEMG contraction burst dataset to assess a different task. For instance, in the field of bio-metrics, subject authentication, identification and verification tasks can be proposed from sEMG data. In this vein, the subjects' identification number can be retrieved from the dataset and used as a label to train and validate a model for any of these proposed tasks.

Chapter 9

Conclusion

The goal of this thesis is to investigate the possibility of classifying the underlying muscles directly from the sEMG signal. This is motivated by the fact that the correct traceability of the implicated muscles in EMG studies is relevant in the periodic evaluation process that is carried out for instance in muscle training program effectiveness for athletes, as well as in routine reviews for muscle rehabilitation. In this vein, this exploratory study proposes the collection of sEMG data from which a selected deep learning-based architecture can be trained and validated.

The proposed data collection protocol along with Delsys Maize Sensor and Delsys Trigno Discover software allowed the collection of 4361 sEMG contraction bursts over 16 channels from 9 different subjects. The collected data was processed, labeled and augmented into 69.776 data samples. Samples were used in the training and validation of the model for the task of muscle classification (left biceps, right biceps, left tibialis anterior and right tibialis anterior).

The state-of-the-art of deep learning models used on sEMG data revealed the use of hybrid approaches that combine effective feature extraction based on CNNs, combined with LSTM networks, which can learn long term sequential dependencies that are inherent in sEMG data. In this vein, the proposed model architecture is defined as an hybrid CNN-LSTM model.

CNN weight initialization via greedy layer wise unsupervised pretraining strategy was carried out for both CNN layers defined in the hybryd CNN-LSTM model architecture. In this strategy, autoencoder architectures were used upon each CNN layer individually for the reconstruction of its inputs. This was achieved by using the current CNN layer as encoder and then attaching a CNN deconvolution layer as decoder. MSE was defined as loss function and ADAM as optimizer over 100 epochs. The achieved minimum loss value for the first AE was 0.0029 and 0.0003 for the second one. The parameters learned in this process were transferred to the final model. The results of using the weight initialization from the previously mentioned strategy demonstrated an increase in the intrapersonal validation accuracy (around 3%) compared to using random weights. However, the accuracy reached in the interpersonal validation dataset remained the same independent of the weight initialization.

The hybrid CNN-LSTM model was trained on 80% of the data from 8 subjects (notice that the leave-one-out subject is not part of the training). The training was carried out using categorical cross entropy as loss function (that is ideal for multi-class classification) along with ADAM as optimizer for 200 epochs. Two validation scenarios were proposed: intrapersonal and interpersonal scenario. The first validation scenario uses the remaining 20% of the data from the 8 subjects and is basically assessing the performance of the model over new bursts from previously seen subjects (scenario in which 95% accuracy is achieved using pretraining weight initialization and 92% when using random weight initialization). This means that the model can accurately generalize and classify muscle contraction bursts, at least for the subjects involved in the training. On the other hand, the second validation scenario uses all contraction bursts from a leave-one-out subject to assess the performance over unseen subjects (scenario in which roughly 48% classification accuracy is achieved independent on the weight initialization). These results indicate that contraction bursts from new subjects are not yet possible to be classified accurately.

Bibliography

[1] Foteini Agrafioti et al. "Heart biometrics: Theory, methods and applications". In: *Biometrics* 3 (2011), pp. 199–216.

[2] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. "Understanding of a convolutional neural network". In: *2017 International Conference on Engineering and Technology (ICET)*. 2017, pp. 1–6. DOI: 10.1109/ICEngTechnol.2017.8308186.

[3] Muhammad Ali et al. "An accurate CT saturation classification using a deep learning approach based on unsupervised feature extraction and supervised fine-tuning strategy". In: *Energies* 10.11 (2017), p. 1830.

[4] Aditya Amberkar et al. "Speech recognition using recurrent neural networks". In: *2018 international conference on current trends towards converging technologies (ICCTCT)*. IEEE. 2018, pp. 1–4.

[5] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling". In: *arXiv preprint arXiv:1803.01271* (2018).

[6] Komal Bajaj, Dushyant Kumar Singh, and Mohd Aquib Ansari. "Autoencoders based deep learner for image denoising". In: *Procedia Computer Science* 171 (2020), pp. 1535–1541.

[7] Dor Bank, Noam Koenigstein, and Raja Giryes. "Autoencoders". In: *arXiv preprint arXiv:2003.05991* (2020).

[8] Tianzhe Bao et al. "A CNN-LSTM hybrid model for wrist kinematics estimation using surface electromyography". In: *IEEE Transactions on Instrumentation and Measurement* 70 (2020), pp. 1–9.

[9] Tanvi Bhandarkar et al. "Earthquake trend prediction using long short-term memory RNN." In: *International Journal of Electrical & Computer Engineering (2088-8708)* 9.2 (2019).

[10] David Charte et al. "A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software and guidelines". In: *Information Fusion* 44 (2018), pp. 78–96.

[11] Xiang Chen et al. "Hand gesture recognition based on surface electromyography using convolutional neural network with transfer learning method". In: *IEEE Journal of Biomedical and Health Informatics* 25.4 (2020), pp. 1292–1304.

[12] Kartik Chitturi and Peter Onyisi. "How easily can neural networks learn relativity?" In: *Journal of Physics: Conference Series*. Vol. 1085. 4. IOP Publishing. 2018, p. 042020.

[13] Delsys. *Delsys Trigno Maize, Base Station and Trigno Discover Software*. [Online; accessed June, 2022]. 2022. URL: https://delsys.com/trigno-maize/.

[14] Delsys. *Delsys Trigno Maize, Sensor User's Guide*. [Online; accessed January,2022]. 2022.

[15] Roger M. Enoka. "Morphological Features and Activation Patterns of Motor Units". In: *Journal of Clinical Neurophysiology* 12 (1995), pp. 538–559.

[16] Hayri Ertan and Ismail Bayram. "Chapter 3 - Fundamentals of human movement, its control and energetics". In: *Comparative Kinesiology of the Human Body*. Ed. by Salih Angin and Ibrahim Engin Şimşek. Academic Press, 2020, pp. 29–45. ISBN: 978-0-12-812162-7. DOI: https://doi.org/10.1016/B978-0-12-812162-7.00003-5. URL: https://www.sciencedirect.com/science/article/pii/B9780128121627000035.

[17] María García-Ordás et al. "Detecting Respiratory Pathologies Using Convolutional Neural Networks and Variational Autoencoders for Unbalancing Data". In: *Sensors* 20 (Feb. 2020). DOI: 10.3390/s20041214.

[18] Sujan Ghimire et al. "Deep solar radiation forecasting with convolutional neural network and long short-term memory network algorithms". In: *Applied Energy* 253 (2019), p. 113541.

[19] Bacary Goudiaby, Alice Othmani, and Amine Nait-ali. "Eeg biometrics for person verification". In: *Hidden Biometrics*. Springer, 2020, pp. 45–69.

[20] Simon Haykin. *Neural networks and learning machines, 3/E*. Pearson Education India, 2009.

[21] Zheng Hu, Jiaojiao Zhang, and Yun Ge. "Handling vanishing gradient problem using artificial derivative". In: *IEEE Access* 9 (2021), pp. 22371–22377.

[22] Naveen Kumar Karnam et al. "EMGHandNet: A hybrid CNN and Bi-LSTM architecture for hand activity classification using surface EMG signals". In: *Biocybernetics and Biomedical Engineering* 42.1 (2022), pp. 325–340.

[23] Mohamed Kazamel and Paula Province Warren. "History of electromyography and nerve conduction studies: A tribute to the founding fathers". In: *Journal of Clinical Neuroscience* 43 (2017), pp. 54–60.

[24] Jun Kimura. "Electrodiagnosis in diseases of nerve and muscle: principles and practice". In: (2013).

[25] Mrigank Krishan et al. "Air quality modelling using long short-term memory (LSTM) over NCT-Delhi, India". In: *Air Quality, Atmosphere & Health* 12.8 (2019), pp. 899–908.

[26] Weibo Liu et al. "A survey of deep neural network architectures and their applications". In: *Neurocomputing* 234 (2017), pp. 11–26.

[27] Marco Maggipinto et al. "A convolutional autoencoder approach for feature extraction in virtual metrology". In: *Procedia Manufacturing* 17 (2018), pp. 126–133.

[28] Sainik Kumar Mahata, Dipankar Das, and Sivaji Bandyopadhyay. "Mtil2017: Machine translation using recurrent neural network on statistical machine translation". In: *Journal of Intelligent Systems* 28.3 (2019), pp. 447–453.

[29] Batta Mahesh. "Machine learning algorithms-a review". In: *International Journal of Science and Research (IJSR).[Internet]* 9 (2020), pp. 381–386.

[30] Núria Massó et al. "Surface electromyography applications". In: *Apunts: Medicina de l'esport; Vol.: 44 Núm.: 166* 45 (Jan. 2010).

[31] Ganesh R Naik. "Computational intelligence in electromyography analysis: a perspective on current applications and future challenges". In: (2012).

[32] Christopher Olah. *Understanding LSTM Networks – colah's blog*. 2015. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/#fnref1.

[33] Marco Pedersoli et al. "Areas of attention for image captioning". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 1242–1250.

[34] PLUX wireless biosignals. *Contraction Burst Detection Algorithm, Biosignals Notebooks Repository*. [Online; accessed February,2022]. 2022. URL: https://github.com/biosignalsplux/biosignalsnotebooks/blob/master/biosignalsnotebooks/biosignalsnotebooks/detect.py.

[35] PLUX wireless biosignals. *OpenSignals, Biosignals Notebooks Python Package*. [Online; accessed February,2022]. 2022. URL: https://pypi.org/project/biosignalsnotebooks.

[36] Susanna Rampichini et al. "Complexity analysis of surface electromyography for assessing the myoelectric manifestation of muscle fatigue: A review". In: *Entropy* 22.5 (2020), p. 529.

[37] Mamun Bin Ibne Reaz, M Sazzad Hussain, and Faisal Mohd-Yasin. "Techniques of EMG signal analysis: detection, processing, classification and applications". In: *Biological procedures online* 8.1 (2006), pp. 11–35.

[38] Mónica Rojas-Martıánez et al. "High-density surface electromyography signals during isometric contractions of elbow muscles of healthy humans". In: *Scientific data* 7.1 (2020), pp. 1–12.

[39] Leonard Sabetti and Ronald Heijmans. "Shallow or deep? Training an autoencoder to detect anomalous flows in a retail payment system". In: *Latin American Journal of Central Banking* 2.2 (2021), p. 100031.

[40] Abhinav Sagar. "Generate High Resolution Images With Generative Variational Autoencoder". In: *arXiv preprint arXiv:2008.10399* (2020).

[41] Miguel Simão, Pedro Neto, and Olivier Gibaru. "EMG-based online classification of gestures with recurrent neural networks". In: *Pattern Recognition Letters* 128 (2019), pp. 45–51.

[42] Vivienne Sze et al. "Efficient processing of deep neural networks: A tutorial and survey". In: *Proceedings of the IEEE* 105.12 (2017), pp. 2295–2329.

[43] Muhammad Zahak Jamal. *The Motor Unit Structure*. 2012. URL: https://www.intechopen.com/chapters/40131.

Consent form for disclosing Master's thesis

Student:	Esteban VELASQUEZ RENDON
Enrolment number:	0571096
Study programme:	MSc. Applied Computer Science
Academic year:	2022

Master's thesis

Title:	Muscle Classification Via Hybrid CNN-LSTM Architecture and sEMG Signals
Supervisor:	Lubos Omelina

Any Master's thesis for which the student has obtained a result of 10/20 or more, and for which no non-disclosure agreement (NDA) was drawn up, can be included at no charge in the Vubis catalogue of the University Library as long as the student has given their prior explicit consent.

In the context of the possibility of making their Master's thesis available free of charge, the studentchooses one of the following options:

- ☒ OPEN ACCESS: worldwide access to the full text of the Master's thesis
- ☐ ONLY FROM THE CAMPUS: access to the full text of the Master's thesis is only possible from the VUB network
- ☐ EMBARGO FOLLOWED BY OPEN ACCESS: worldwide access to the full text of the Master's thesis only after a specific date, namely Click or tap here to enter text.
- ☐ EMBARGO FOLLOWED BY ACCESS ONLY FROM THE CAMPUS: access to the full text of the Master's thesis only from the campus and only after a specific date, namely Click or tap here to enter text.
- ☐ FULL TEXT NEVER ACCESSIBLE: no access to the full text of the Master's thesis

The supervisor confirms acknowledgement of the intention of the student to make the Master'sthesis available in the Vubis catalogue of the University Library.

Date: 28/08/2022

Signature of supervisor:

This document will be included in the Master's thesis. Any student who fails to include the document in their Master's thesis and/or has failed to indicate a choice and/or has failed to sign the document and/or has failed to inform the supervisor will be considered as not having granted permission to make their thesis public; in that event, the Master's thesis will only be archived and will not be accessible to the public.

Drawn up at on 28/08/2022

Signature of student

ANNEX:

GITHUB REPOSITORY WITH MASTER THESIS CODE

https://github.com/esvelaren/master-thesis-VUB-sEMG