# Convolutional Recurrent Neural Network: IMU-based Locomotion Intent and Gait Phase Prediction for Transfemoral Amputees

Master Research Project

Daniel Marcos Mazón (s3734277)

September 14, 2021

Internal Supervisor(s): Prof. Dr. Raffaella Carloni (Artificial Intelligence, University of Groningen)
Second Supervisor: Prof. Dr. Lambert Schomaker (Artificial Intelligence, University of Groningen)
Third Supervisor: Marc Groefsema, MSc (Artificial Intelligence, University of Groningen)

**Artificial Intelligence**
**University of Groningen, The Netherlands**

**Abstract**

This paper focuses on the design of deep neural network architectures for the real-time prediction of locomotion modes, transitions and gait phases for ten healthy subjects and one osseointegrated transfemoral amputee by using inertial measurement units (IMU). Different neural network configurations are investigated by combining convolutional and recurrent layers. As input to the networks, the frequency aspect in the form of a spectrogram, of one IMU (located in the thigh) or two IMUs (located in both the thigh and the shank) are used. The system is able to predict seven different locomotion modes (sitting, standing, walking, ramp ascent and descent, stair ascent and descent), transitions among this locomotion modes and the gait phases corresponding to each of the locomotion modes. The results show that a system composed of CNN + LSTM networks is able to predict user intention with a mean F1-score of 0.893 and 0.910 for the healthy subjects, and 0.921 and 0.947 for the amputee subject, using one and two IMUs respectively with a 5-fold cross-validation.

# Acknowledgement

I would like to thank Prof. Dr. Raffaella Carloni and Prof. Dr. Lambert Schomaker for the trust they put on me for doing this project as well as for their supervision and contribution. I would also like to thank Marc Groefsema, MSc for his more than useful input and help on the development of this project.

Special thanks to Prof. Dr. Hermie Hermens (Roessingh Research and Development, Enschede, The Netherlands) for the data of the osseointegrated amputee contained in the MyLeg dataset, which is part of the European Commission's Horizon 2020 MyLeg Project.

The author would like to also thank Blair Hu, Elliott J. Rouse and Levi J. Hargrove for the creation and publication of the EN-ABL3S dataset.

Lastly, I would like to thank my family and my friends for the support given during the university years.

# Contents

# List of Figures

# List of Tables

# 1 Introduction

## 1.1 Motor Intent Recognition

In recent years there has been a growing interest in the field of robotic prosthesis, with the aim of improving their efficiency and, ultimately, the quality of life of amputee people that are bound to use them. In this research specifically, we focus on the control of a robotic transfemoral prosthesis and its ability to correctly predict the user's locomotion mode intentions and gait phase with the use of deep learning techniques.

We aim at improving the physical capabilities of amputee people in their daily life, but there are many challenges that need to be overcome to reach this goal. As with any machine, there is a computational load from the gathering of input as well as the calculation of the results. In order for the user not to perceive any lag in the response of the prosthesis and, therefore, to reduce the discomfort of using it, we need to predict the locomotion intention within 300 ms [1]. This way, changing between different locomotion modes can be made in a smooth way without the user taking notice. We pretend to achieve this low computation cost by looking for efficient network architectures as well as proper data preprocessing.

Another challenge we face is replicating the daily locomotion activities that happen in real life, such as walking, ramp ascent and descent or going upstairs and downstairs. Some studies only focus on one type of locomotion activity or recreate said activity using tools such as treadmills [2], which do not quite resemble the real life aspect of these activities. Replicating daily life activities also includes the transitions between different locomotion modes, such as going from walking to downstairs or going upstairs to walking. With this aim, some researches have found solutions that involves the user changing manually between modes [3] with the use of a key fob or some other kind of command. This of course imposes yet another burden for the cognitive load of the user that should be avoided.

With all this into account, in this project we will develop a method for the efficient prediction of user locomotion intention, both for locomotion modes as well as transitions between them, as well prediction of the gait phases of the movement. We will investigate what kind of input data describes better the movement of the user in terms of real-time classification, how many measurement units are necessary to achieve proper prediction accuracy or which network architecture fits better our purpose.

## 1.2 Research Question

The main objective of this project is to create a control architecture for the robotic transfemoral prosthesis so it is capable of predicting the user locomotion intention, that is, stating with as much detail as possible which is the next action of the user given the current information of the movement. For that, we will predict not only the locomotion mode or transition in which the user currently is, but also the gait phase of the leg. Knowing the gait phase can add important information for the control of the prosthesis, as it can better adapt to the movement.

We will test whether the system's performance drops when using one single IMU (inertial measurement unit) placed on the upper leg with respect to using 2 IMUs located on both the upper and lower leg.

Regarding the data we obtain from the IMU we have two different approaches. The first consists on using the raw information coming from the accelerometer and the gyroscope inside the IMU directly into the neural network, without further processing, which can alleviate some computational load from the whole system but might be less interpretative. On the other hand, we can compute the spectrogram of the signals coming form the IMU and use those as input for the networks; the frequency information contained in the spectrogram might help the network discriminate better between classes but at a higher computational cost. We will test both approaches and discussed the results based on their performance.

To sum up, in this project we will try to answer the following research questions:

- Predict user movement by identifying locomotion modes, transitions and gait phases.
- Do the prediction in under 300ms as to not cause any discomfort to the user.
- Compare between various network architectures and number of IMUs.

## 1.3 Scientific Relevance for Artificial Intelligence

With this project we will try to make a contribution to the field of lower limb prosthesis. We want to improve their efficiency, which will have a huge impact on the lives of people using them, as the prosthesis will be able to respond faster and with more accuracy to their locomotion intentions.

We will be investigating movement recognition, which not only has an application on lower limb prosthesis but on other research fields such as surveillance systems, health-care for elderly or rehabilitation among others. The results obtained in this project could potentially be used in these other areas.

This project falls inside the MyLeg[1] project, which is a European-funded project that aims at developing a new generation of powered transfemoral prosthetic legs that can be intuitively operated, sensed, and trusted as a healthy and reliable counterpart for a variety of tasks.

[1] http://www.myleg.eu/

# 2 Theoretical Framework

## 2.1 Neural Networks

In this section it will be explained the main neural networks architectures used in this project as well as their main characteristics and differences among each other. It starts with convolutional neural networks, following with recurrent neural networks, where two different types are analyzed (gated recurrent units and long-short term memory cells), and finishing with the combination of the two in the convolutional recurrent neural networks.

### 2.1.1 Convolutional Neural Networks

The first type of network we talked about in this project are convolutional neural networks. CNNs were first introduced in [4] and, after the appearance of Alexnet from [5], the enthusiasm on them started to rise. This type of networks combined image processing with the classification tasks of neural networks. This type of network contain an input layer, a set of hidden layer including convolutional layers that perform image filtering and help extract features from them, and an output layer. This convolution task is done using a convolutional kernel that slides through the input, creating a feature map that later on is used as input for the subsequent layer. An example of the architecture of a typical convolutional neural network can be seen is Figure 1.

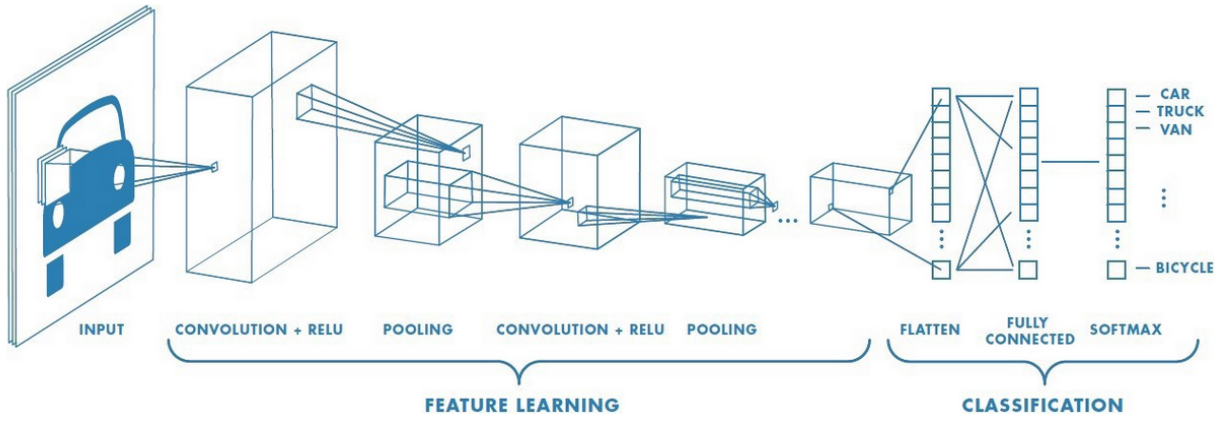



Figure 1: Convolutional Neural Network Architecture. From: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

**Convolutional Layers** As said before, the convolutional layers are in charge of filtering the input image and extracting features, that later can be used for the classification of the given input. For that some hyperparameters must be taken into account, which are the number of kernels, the kernel size and the stride. The values of this hyperparameters must be set artificially and vary depending on the task or type of network. Each convolutional kernel is initialized differently so there are multiple different feature maps.

Figure 2 shows a depiction of the convolution process. An image of size 5x5 (green matrix) is used as input to the convolutional layer. Then, a kernel of size 3x3 (yellow matrix) slides through the image following the stride size, in this case 1, and computing the convolution. For that, the layer performs an element-wise multiplication between the image and the kernel and the sum of the remaining elements is the convolved feature that is added to the final feature map.

**Pooling Layer** After the convolutional layer there is usually a pooling layer, whose purpose is to reduce the dimensionality of the feature map and thus the computation requirement of the neural network. There are two different types of pooling: max-pooling and average-pooling. In max-pooling, the maximum value from within the kernel area is taken while in average-pooling the average of all values withing the kernel area is considered. After this two layers, convolutional and pooling, features are finally extracted and the network is ready for the classification.

**Dense Layer** The dense layers at the end of the neural network allow the learning of non-linear combinations from the features computed by the convolutional layers. This output is flattened and then inputed into the fully connected layer, inn which the classification will be learnt after a series of training epochs. This part of the neural networks is usually ended with a softmax layer, which is the main tool for classification tasks.

### 2.1.2 Recurrent Neural Networks

In some cases, different sequences that are inputed into the network share information between them due to some temporal dependencies, like in fields such as speech and handwriting recognition or movement prediction, which is the case at study in this project. For this kind of problems, recurrent neural networks appeared since they are able to propagate previous information through all the network, since the output of one layer depends directly on the current input as well as in all previous inputs. In spite of this advantage, if the RNN gets too long, the vanishing gradient problem might

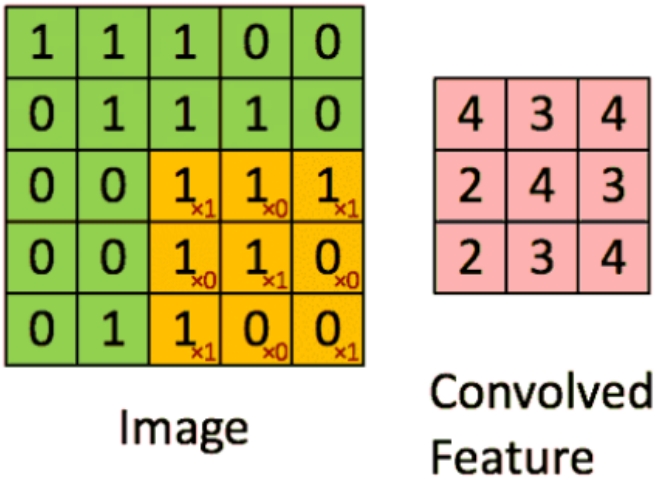



Figure 2: Result of a applying a convolution kernel of 3x3 to an input image of 5x5, obtaining an output feature map of 3x3. From: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

appear, by which more recent information becomes less important. Figure 3 shows the architecture of an unfolded RNN.

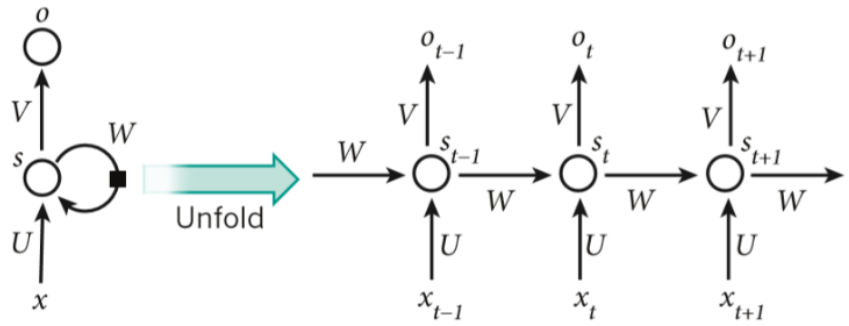



Figure 3: Example of an unfolded RNN architecture, the input of the network at state $s_t$ is dependent of the previous output of state $s_{t-1}$ and input $x_t$. (image from [6])

**Long-Short Term Memory Cells** To tackle the vanishing gradient problem, the LSTM architecture appeared. It was first proposed by [7] to solve the log term dependency problem previously discussed and in [8] it is found an explanation of the LSTM module.

As seen in Figure 4, the cell state resembles a conveyor belt by which the information that is inputted flow through, and information is added or removed by means of some linear interactions

with the so called gates, namely the forget gate, the input gate and the output gate.

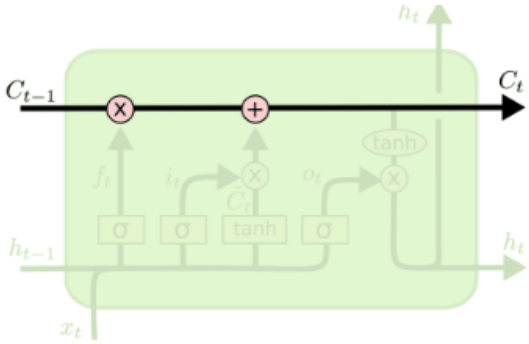
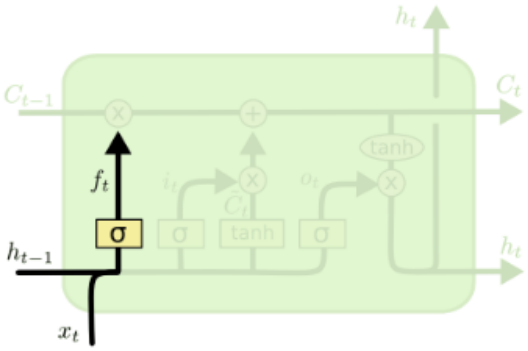



Figure 4: Information flowing through an LSTM cell in order to be update by the forget, input and output gates. (image from [8])

The forget gate (Figure 5) is in charge of deciding which information must be removed from the cell state. It is implemented with a sigmoid function that takes into account the current input $x_t$ and previous information $h_{t-1}$ and outputs 1 in order to keep the information or 0 in order to get rid of it.

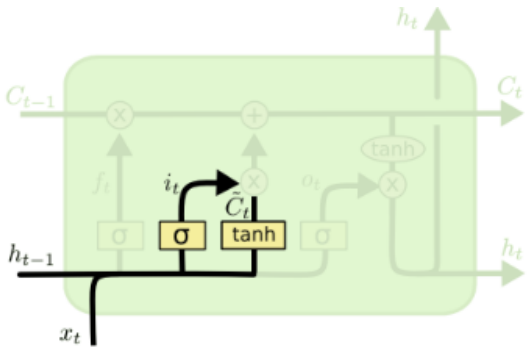



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 5: Forget gate architecture in an LSTM cell. (image from [8])

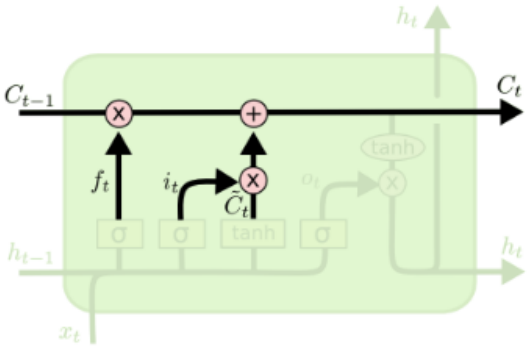
Figure 6 shows the second step, or second gate, which is the input gate. This refers to what kind of information it is going to be kept in the cell state. A sigmoid functions decides which values will be updated and a tanh layer creates the new values for the cell state.

After that, as seen in Figure 7, the old state is updated by multiplying it by $f_t$, what needs to be forgotten, and than it is added $i_t * C_t$ in order to finish updating the cell value.




$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

Figure 6: Input gate architecture in an LSTM cell. (image from [8])




$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 7: Updating the previous state of the LSTM cell to the new infomration. (image from [8])

Finally, the output gate (Figure 8) decides what is going to be outputted. A sigmoid layer decides what it is going to output and a tanh layer to output the parts that were decided.

**Gated Recurrent Neural Network** A variant architecture to the LSTM is the Gated Recurrent Neural Network (GRU), first proposed in [9]. The main difference of GRU with respect to LSTM is that now the forget gate and the input gate are combined together into an update gate that controls which information should be taken into account, and the output gate now changes into a reset gate. Figure 9 shows the architecture of the gated recurrent unit and all its gate update computations.

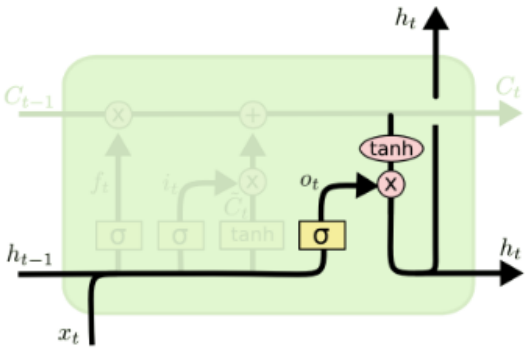



Figure 8: Output gate architecture in an LSTM cell. (image from [8])

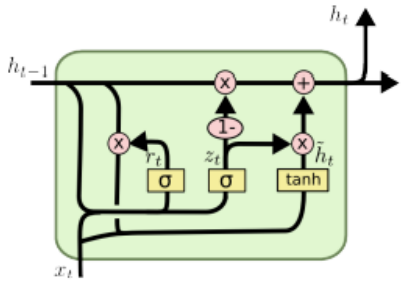



Figure 9: Architecture of a gated recurrent unit and its gate computations. (image from [8])

### 2.1.3 Convolutional Recurrent Neural Networks

In this project it is made use of the so called Convolutional Recurrent Neural Networks, which are a special type of neural network that combines the main features from convolutional networks and recurrent networks. That is, the visual learning from CNNs and the sequence learning of RNNs. The architecture of a convolutional recurrent neural networks, shown in Figure 10, starts with an input layer that receives sequential images that are first process by the convolutional layer, extracting visual features. These visual features are then analyzes by the recurrent layers, which could be LSTM, GRU or other type of recurrent cells in order to extract temporal features.

## 2.2 Movement prediction and recognition

Past studies have focused only on the detection of the gait phases, achieving high detection accuracies on it. In [11] and [12], heel strike and toe events are detected from both healthy subjects and transfemoral amputees using a single IMU. Both studies use thresholding on the IMU signals to

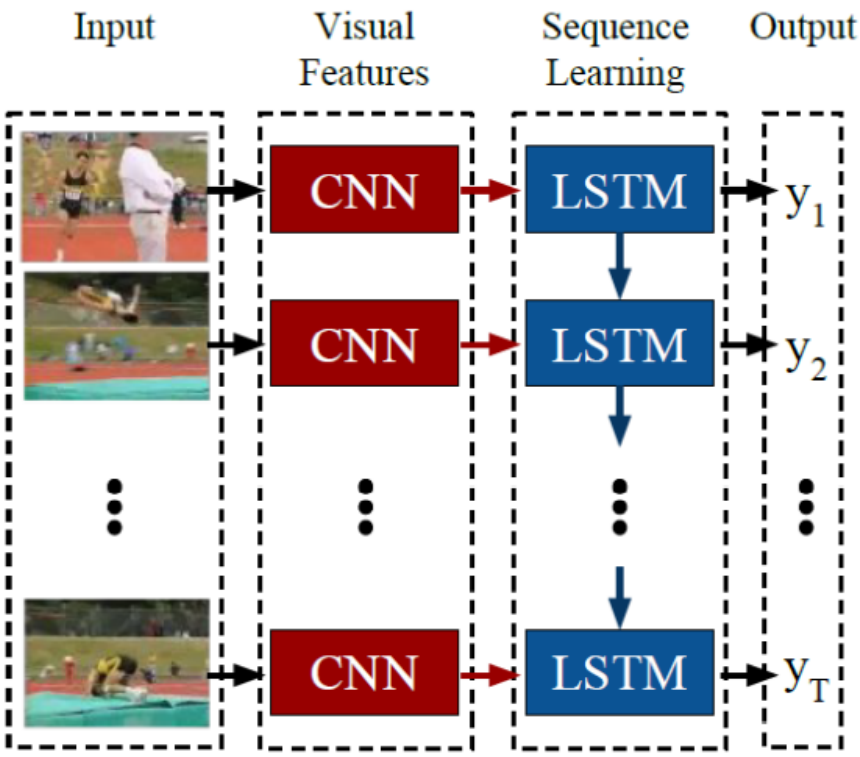



Figure 10: Architecture of a recurrent neural network making use of LSTM cells. (image from [10])

achieve 100% and 99.78% detection accuracy respectively in both the healthy and the transfemoral amputee subjects. In [13], six gait phases, such as loading response, mid-stance, or swing are recognized. twelve healthy individuals took part and walking data was recorded by seven IMUs located in different parts of the lower body. Raw information from the IMUs was fed into a CNN, obtaining 95.1% accuracy. In [14], two IMUs are used for gait phase detection of stroke survivors. Different features from the sensors are extracted and Quadratic Discriminant Analysis (QDA) is used to obtain a final accuracy of 96.5%.

With respect to locomotion modes and transitions classification, [15] and [16] focus on the recognition of only locomotion modes. Specifically, in [15] raw data from one IMU is fed into a CNN obtaining an overall accuracy of 87.62%. [16] uses a Recurrent Neural Network (RNN) based on Long Short-Term Memory (LSTM) cells to achieve 96.63% recognition accuracy on multiple locomotion activities using two IMUs located in the arm. If we take a look into the transition classification, [17] presents a multi-level architecture for both the recognition (accuracy of 99% and 96%) and prediction (accuracy of 99% and 93%) of locomotion modes and transitions respectively,

using features extracted from IMU data and classifying with a Support Vector Machine (SVM) on ten able-bodied subjects. On the other hand, [18] presents a single CNN in charge of the classification and using as input the spectrogram representation of multiple IMUs located in different parts of the leg, achieving 7.75% overall error rate with individual error rates of 5.45% and 18.08% for locomotion modes and transitions respectively. [19] performed a research using three IMUs and a CNN on ten able-bodied subjects and one amputee subject to obtain 94.15% and 89.23% accuracy respectively. In [20] it is shown how the mix of a capacitive sensing system and mechanical sensors can improve the performance of the classification. By using IMU and a load cell, they are able to obtain 95.8% average recognition accuracy with an SVM classifier and 94.9% accuracy with QDA on a study with six transfemoral amputees. In [21], three able-bodied and two amputee subjects take part in the research and an IMU and two pressure sensors are used with Hidden Markov Models (HMM) to predict the user intention with an accuracy of 95.8%. In [22], it is shown how a network such as Wavenet achieves a peak F1-score of 87.17% in the case of one IMU, and a peak of 97.88% in the case of two IMUs, while using features in the time-domain with a dataset containing information of ten able-bodied subjects, for the classification of different locomotion modes. Additionally, [23] used time-domain and frequency-domain features to predict locomotor and transition intentions. Features are inputed into a recurrent neural network to achieve a mean F1-score of 84.77% and 93.06% with one and two IMUs respectively.

Lastly, we can find studies mixing gait phases as well as locomotion and transition classification such as [24] and [25]. In [24] we see a similar multi-level architecture as the one presented in [17], this time including a level for the recognition of gait phases. Three transtibial amputees took part in the project, and data from six locomotion modes and ten transitions was obtained using two IMUs and a load cell. The different classifications were done using a mix of QDA and thresholding on computed features of the raw data from the sensors, obtaining 93.21% recognition accuracy with no missing transitions. In [25], seven able-bodied subjects take part and the multi-level architecture is repeated, this time with Linear Discriminant Linear Analysis (LDA) classifier and features computed from three IMUs and pressure insoles, achieving a final accuracy of 99.71% with no missed transitions.

In Table 1 it is presented a summary of the main contributions of machine learning and deep learning techniques for the classification of locomotion modes, transitions and gait phases. Performance is reported in terms of accuracy and F1-scores, IMU placement in the subject's body, used techniques and whether testing is done in healthy and/or impaired subjects.

In this study, a multi-level architecture composed by multiple neural networks to predict gait phases as well as locomotion modes and transitions is proposed. Multiple neural networks architectures such as CNN and CNN combined with LSTM and GRU layers are investigated. As inputs the spectrogram computed from the IMU data is used, both with one IMU (placed at the thigh) and two IMUs (placed at the thigh and shank). The system is trained to recognize seven locomotor intentions (sitting, standing, level ground walking, stair ascent and descent and ramp ascent and descent), the different transitions among them (twelve transitions in the EN-ABL3S dataset and nineteen transitions in the MyLeg dataset) and the gait phases, which vary depending on the locomotion mode. This study shows that a multi-level architecture made of various CNN-LSTM neural networks can achieve a mean F1-score of 0.893 ± 0.006 using one IMU and 0.910 ± 0.007 using two IMUs with ten healthy subjects [26], and a mean F1-score of 0.921 ± 0.006 using one IMU and 0.947 ± 0.005 using two IMUs with one osseointegrated transfemoral amputee. The main contributions of this paper are:

- Design a multi-level architecture made of different neural networks for the prediction of locomotion modes, transitions and gait phases.
- Use data only available from either one IMU or two IMUs, using their frequency information as input to the neural network architecture.
- Validate the results in two different datasets, one with ten healthy subjects and the other with one transfemoral amputee.
- Obtain a top F1-score of 0.910 ± 0.007 with two IMUs for the healthy subjects and 0.947 ± 0.005 using two IMUs for the amputee subject with a 5-fold cross-validation.

The remainder of this paper is organized as follows. Section 3 explains the datasets that are used in this project and the necessary methods to process the data before it can be used for the training process, Section 4 explains the system architecture as well as the different network configurations to be used, Section 5 present all the results obtained in the proposed experiments while Section 6 discuss these results on the scope of this study and in comparison with the current literature. Finally, Section 7 gives a final overview of the achievements of this work.

| Article | Method | Features | Mean Accuracy | IMU Placement | Classes | Subject(s) |
|---|---|---|---|---|---|---|
| **Gait Phases Classification** | | | | | | |
| [11] 2018 | Thresholding<br>LDA, QDA | 1 IMU time domain | 100% | Lower Leg | Gait Phases | 10 Healthy<br>5 Transfemoral Amputees |
| [12] 2016 | Thresholding | 1 IMU time domain | 99.78% | Lower Leg | Gait Phases | 4 Healthy<br>1 Transfemoral Amputee |
| [13] 2016 | CNN | 6 IMU time domain | 95.1% | Lower Leg | 6 Gait Phases | 12 Healthy |
| [14] 2018 | QDA | 2 IMU time domain | 96.5% | Upper Leg<br>Lower Leg | Gait Phases | 3 Stroke survivors |
| **Locomotion Modes and Transition Classification** | | | | | | |
| [15] 2020 | CNN | 1 IMU time domain | 87.62% | Lower Leg | 5 Locomotions | 30 Healthy |
| [16] 2018 | RNN<br>LSTM | 2 IMU time domain | 96.63% | Upper Arm | 5 actions | 11 Healthy |
| [17] 2020 | SVM | 3 IMU time domain | 96% | Upper Leg<br>Lower Leg | 5 Locomotions<br>All Transitions | 10 Healthy |
| [18] 2020 | CNN | 3 IMU, 2 EMG, 3 GONIO<br>frequency domain | 1.1% (Error) | Upper Leg<br>Lower Leg | 5 Locomotions<br>8 Transitions | 10 Healthy |
| [19] 2019 | CNN | 3 IMU time domain | 94.15%<br>89.23% | Upper Leg<br>Lower Leg | 5 Locomotions<br>8 Transitions | 10 Healthy<br>1 Transfemoral Amputee |
| [20] 2017 | SVM, QDA | 2 IMU, load cell<br>time domain | 95.8% | Lower Leg | 5 Locomotions<br>8 Transitions | 6 Transfemoral Amputees |
| [21] 2019 | HMM | 1 IMU 2 pressure sensor<br>time domain | 95.8% | Lower Leg | 5 Locomotions | 3 Healthy<br>2 Transfemoral Amputees |
| [22] 2020 | Wavenet | 2 IMU time domain | 97.88% (F1-score) | Upper Leg<br>Lower Leg | 7 Locomotions | 10 Healthy |
| [23] 2021 | RNN | 2 IMU time domain<br>frequency domain | 93.06% (F1-score) | Upper Leg<br>Lower Leg | 8 Locomotions<br>24 Transitions | 1 Transfemoral Amputee |
| **Locomotion Modes, Transition and Gait Phases Classification** | | | | | | |
| [24] 2018 | QDA<br>Thresholding | 2 IMU and load cell<br>time domain | 93.21% | Lower Leg | 6 Locomotions<br>10 Transitions<br>Swing and Stance | 3 Transtibial Amputees |
| [25] 2014 | LDA | 3 IMU and pressure insole<br>time domain | 99.71% | Upper Leg<br>Lower Leg | 6 Locomotions<br>10 Transitions<br>Swing and Stance | 7 Healthy |

Table 1: State of the art of for related project from previous years, divided in 3 categories: gait classification, locomotion modes and transition classification, and locomotion modes, transition and gait phases classification.

# 3 Materials

This Section presents the two datasets used in this study as well as the methods to extract the sequences used as input for the deep neural networks.

## 3.1 Data-set

### 3.1.1 EN-ABL3S Dataset

This dataset is a publicly available dataset called Encyclopedia of Able-bodied Bilateral Lower Limb Locomotor Signals (EN-ABL3S) [26], which contains IMU data, both accelerometer and gyroscope data, from ten able-bodied subjects. The data was gathered from seven males and three females with an average age of 25.5 ± 2 years, a height of 174 ± 12 cm, and weight 70 ± 14 kg. From this dataset, only data from the IMUs located in the thigh and the shank is used. This IMU data is sampled at 500 Hz by means of the IMUs MPU-9250 (InvenSense, San Jose, CA, USA).

The available locomotion modes are sitting (S), standing (ST), level ground walking (W), stair ascent (SA) and descent (SD) and ramp ascent (RA) and descent (RD). Each subject was asked to perform two types of circuit in order to record the data. Specifically, the odd trial is: S $\rightarrow$ St $\rightarrow$ W $\rightarrow$ SA $\rightarrow$ W $\rightarrow$ RD $\rightarrow$ W $\rightarrow$ St $\rightarrow$ S; while the even trial is: S $\rightarrow$ St $\rightarrow$ W $\rightarrow$ RA $\rightarrow$ W $\rightarrow$ SD $\rightarrow$ W $\rightarrow$ St $\rightarrow$ S. The stairs consist of four steps and the ramps have slopes of 10°.

The dataset also has information to obtain the transitions between the locomotion modes, as well as the gait phases of each locomotion mode by means of toe off and heel strike events. Figure 11 shows the gait phases intervals for each of the locomotion modes. Gait phases intervals vary depending on the locomotion mode they are linked to and locomotion modes such as sitting and standing do not have gait phase information since they are static modes [27][28][29][30].

Regarding the data labelling, in the original dataset only locomotion modes are labelled. To obtain the transitions, a 500 ms window centered at a transition point (the time step in between two subsequent locomotion modes) is created and the samples inside are labelled according to the transition (i.e., transition from walking to sitting is labelled as W - S). This means that for each transition point, 250 transition samples are obtained.

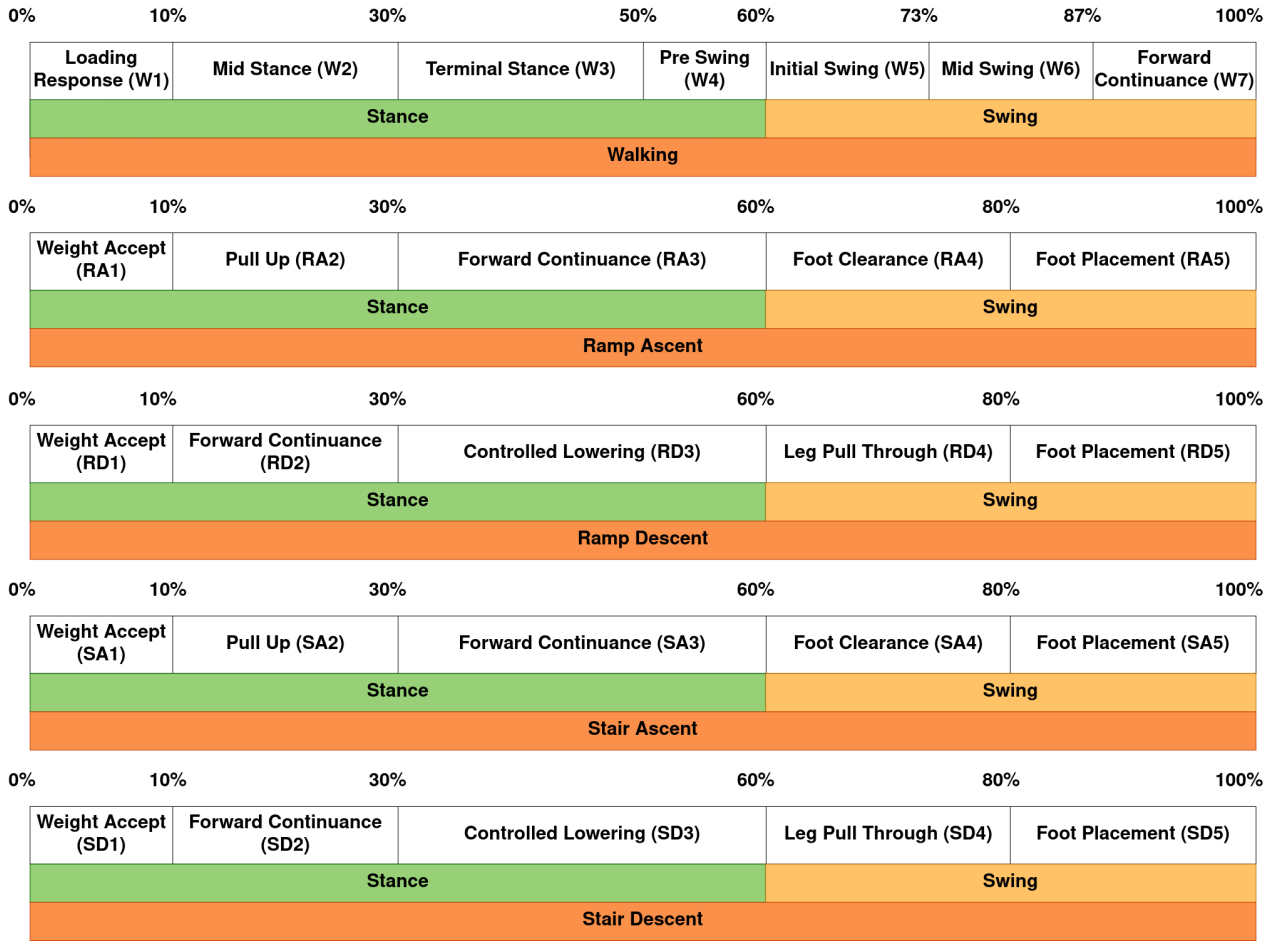



Figure 11: Locomotion modes and corresponding gait phases.

### 3.1.2 My-Leg Dataset

This dataset was collected at the Roessingh Research and Development center (Enschede, The Netherlands) on one osseointegrated transfemoral amputee subject (male, 75 years old, 84.1 kg, 186.6 cm, left-sided amputation since 45 years, osseointegration since 4 years, functional level K3), using a 3R80 Ottobock prosthetic knee (`www.ottobockus.com`) and a Variflex Össur prosthetic ankle (`www.ossur.com`). The data were collected from the subject by using wearable electromyographic sensors and eight IMUs as part of the Xsens MVN Link motion capture system (Xsens Technologies B.V., The Netherlands, `www.xsens.com`)[2]. In this study, data from one IMU in the thigh and another IMU located in the shank is used. The IMU data is sample at a frequency of 1000 Hz.

[2]The study, under protocol number NL67247.044.18, was evaluated and approved by the Medical Ethics Review Commitee of the University of Twente (The Netherlands) on December 13, 2018.

The subject was asked to do seven different locomotion modes: sitting (S), standing (ST), level ground walking (W), stair ascent (SA) and descent (SD) and ramp ascent (RA) and descent (RD) on a circuit. The ramps have a slope of 10°for three meters, and continue on with a slope of 15°. Regarding the data labelling, in the original dataset only locomotion modes are labelled. To obtain the transitions, a 500 ms window centered at a transition point (the time step in between two subsequent locomotion modes) is created and the samples inside are labelled according to the transition (i.e., transition from walking to sitting is labelled as W - S). This means that for each transition point 500 transition samples are obtained.

In this dataset there is no information about the gait phases of the subject.

## 3.2 Data Processing

### 3.2.1 Sequence extraction

The inputs to the deep neural networks are sequences, conformed by sequential samples. Each sample contains information from the IMU; in the case of one IMU it has information from one triaxial accelerometer and one triaxial gyroscope for a total of six features, while in the case of two IMUs there is information from two triaxial accelerometers and two triaxial gyroscopes for a total of 12 features. The window size for the sequences is 1.3 seconds with a sliding window of 50 ms. This means that for the EN-ABL3S dataset, a sequence contains 650 samples while, for the MyLeg dataset, a sequence contains 1300 samples

When extracting sequences from the original labelled samples there are three different situations that can happen. One, the extracted sequence falls completely in one locomotion mode or transition, in which case it gets labelled as such. Two, the sequence falls in between a locomotion mode and a transition, getting in this case the labelled of the majority of the samples contained in said sequence. Three, the sequence is in between a transition and a locomotion mode, in which case it gets labelled as the locomotion mode. Figure 12 depicts an example of multiple sequence extractions. The yellow sequences represent case one, in which the extracted sequence falls entirely into one locomotion mode or transition and thus, it is labelled as such. Green sequences represent the second case: the top sequence contains more samples from the walking locomotion mode and is labelled as walking, while the bottom sequence has more samples from the transition Walking to Sitting and so it is labelled as 'W - S'. The last case is represented by the blue sequence. Since this sequence starts in a transition and finishes in a locomotion mode, it is labelled as the locomotion mode.

Regarding gait phases, only on the EN-ABL3S dataset, they are obtained following the gait cycles as established in [27][28][29][30] and following the intervals shown in Figure 11.

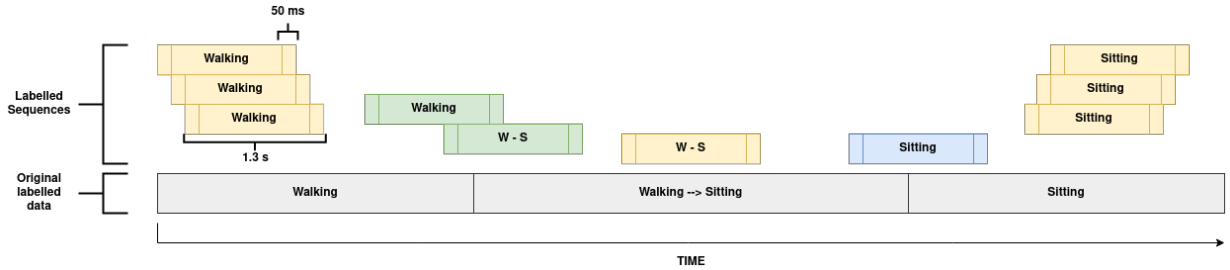



Figure 12: Sequence extraction diagram.

Tables 2, 3 and 4 show the sequence count for each of the used locomotion modes, transitions and gait phases for the respective datasets.

| Label | # of sequences | Label | # of sequences |
|---|---|---|---|
| S | 6639 | SA-W | 257 |
| W | 10009 | W-RD | 272 |
| RA | 3284 | RD-W | 124 |
| RD | 3987 | W-ST | 531 |
| SA | 1304 | ST-S | 533 |
| SD | 1307 | W-RA | 215 |
| ST | 4424 | RA-W | 262 |
| S-ST | 525 | W-SD | 258 |
| ST-W | 128 | SD-W | 236 |
| W-SA | 262 | | |

Table 2: Locomotion modes and transition samples count for En-ABL3S.

### 3.2.2 Image encoding

Given that the human movement has a periodic nature, frequency information is extracted from the IMU data as previously done in [22] and [23]. For that, the raw information from the IMUs is encoded using a spectrogram. First of all, the Short Time Fourier Transform (STFT) is computed to obtain the frequency-domain information from time-series data.

| Label | # of sequences | Label | # of sequences |
|---|---|---|---|
| W1 | 937 | RD3 | 682 |
| W2 | 1813 | RD4 | 981 |
| W3 | 2049 | RD5 | 1042 |
| W4 | 1365 | SA1 | 150 |
| W5 | 1531 | SA2 | 336 |
| W6 | 1724 | SA3 | 547 |
| W7 | 2128 | SA4 | 285 |
| RA1 | 361 | SA5 | 243 |
| RA2 | 718 | SD1 | 141 |
| RA3 | 932 | SD2 | 364 |
| RA4 | 677 | SD3 | 203 |
| RA5 | 858 | SD4 | 325 |
| RD1 | 357 | SD5 | 510 |
| RD2 | 1049 | | |

Table 3: Number of sequences for the gait phases in the EN-ABL3S dataset when considering all possible gait phases. W1 corresponds to gait phase 1 of walking and so on, according to figure 11.

$$STFT(x[n];w,k) = \sum_{\infty}^{n=-\infty} x[n]w[n-k]e^{-j\omega n} \tag{1}$$

After squaring the output signal, the spectrogram is modified with a non-linear scaling called mel scale, which has already proved useful in similar classification tasks [18]. The mel scale is known to help amplify the lower frequencies, as shown in figure 13, which is were most of the information for human movement can be found. This scale can be computed as [31]:

$$2595 \cdot \log_{10}(1 + \frac{f}{700}) \tag{2}$$

where 2595 is a constant value ensuring that 1000 Hz correspond to 1000 mel and 700 is the corner frequency at which the scales changes from linear to logarithmic.

This signal is then converted into dB and normalized in the range [0, 1] so it can be processed by the neural networks. For the calculation of the STFT, a Hann window of size 20 with an offset of 13 is used. When using the mel scale, the Hz scale is partitioned in 10 bins in order for some channels in the mel spectrogram not to return an empty response. To implement all this process the *Python* package called LibROSA [32] is used.

| Label | # of sequences | Label | # of sequences |
|---|---|---|---|
| S | 5963 | ST-RA | 200 |
| ST | 4978 | ST-RD | 179 |
| W | 4182 | W-SA | 537 |
| SA | 2736 | W-RA | 186 |
| SD | 2225 | W-S | 291 |
| RA | 1612 | W-SD | 193 |
| RD | 2208 | W-ST | 147 |
| S-W | 136 | SA-ST | 71 |
| S-ST | 536 | SA-W | 521 |
| ST-SD | 474 | SD-ST | 77 |
| ST-SA | 71 | SD-W | 513 |
| ST-S | 45 | RA-RD | 427 |
| ST-W | 3164 | RD-W | 483 |

Table 4: Number of sequences for locomotion modes and transitions in the MyLeg dataset.

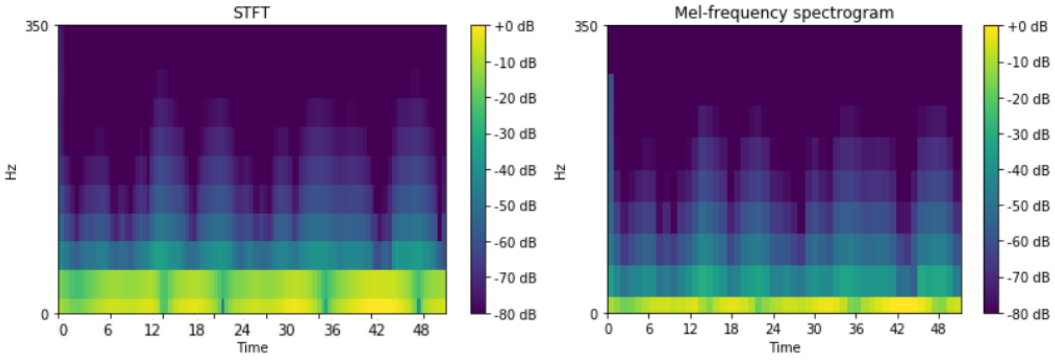



Figure 13: Initial spectrogram (left) and mel spectrogram (right) from one of the signals from the IMUs. Higher frequencies are attenuated after the mel scaling.

# 4 Methods

In this Section, the overall system architectures as well as the different architectures considered for the individual neural networks are discussed, based on the previous work done in [22] and [23]. In the end, an overview of the experimental setting is given.

## 4.1 System Architecture

In this Section it is explained the system architecture that this study follows for the prediction of locomotion modes, transitions and gait phases from the input sequences.

This study proposes a multi-level architecture as depicted in Figure 14. The input, the spectrogram of the input sequence, goes into a first level classification (blue box in figure 14). This first level classifies to which locomotion mode the input sequence belongs to. It is composed of one single neural network and here, locomotion mode and transition sequences are treated alike. This means, for example, that sequences labelled as the walking locomotion mode, and sequences labelled as transitions that start in walking, are labelled as walking. This level works as a pre-classification step for the next level.

The second level is made of two parts: one in charge of classifying locomotion modes and transitions (level 2A), and another one in charge of classifying gait phases (level 2B).

Level 2A is composed of 7 different neural networks, each one in charge of a different locomotion mode. Depending on the result of level 1, the input sequence will go into one or another. Locomotion modes and transitions are treated equally in level 1, but in level 2A they get classified as either locomotion mode or one of the transitions. For example, a walking to standing transition that was classified as walking by level 1 would be classified as W - St in level 2A. In the case of having a walking sequence classified as walking by level 1, it would be classified as W - W by level 2A, thus making the distinction from locomotion modes and transitions.

Level 2B is composed of 5 different networks, one for each locomotion mode that has gait phases (sitting and standing are excluded here). The input sequence will go into one network or another depending on the result from level 1, independently of being a locomotion mode or a transition. Note that for the MyLeg dataset, level 2B is ignored since there are not any gait phases to be classified.

Figure 15 shows the process an input sequence undergoes through the system. Two input sequences are shown: one sequence corresponding to the walking locomotion mode with gait phase W1 (red sequence) and another sequence corresponding to a walking to standing transition with gait phase W6 (orange sequence). In the fist level both of them are classified as WALKING. In level 2A, the sequences will go into the ”Walking Locomotion/Transitions Network”, being classified as W to W (red sequence) and W to ST (orange sequence). Something similar will happen in level 2B; both sequences will enter the ”Walking Gait Phases Network” and will be classified as W1 ( red sequence) and W6 (orange sequence).

The final results will be ”W to W - W1” for the red sequence and ”W to ST - W6” for the orange sequence. This procedure will be repeated for any combination of locomotion mode/transition and gait phase.

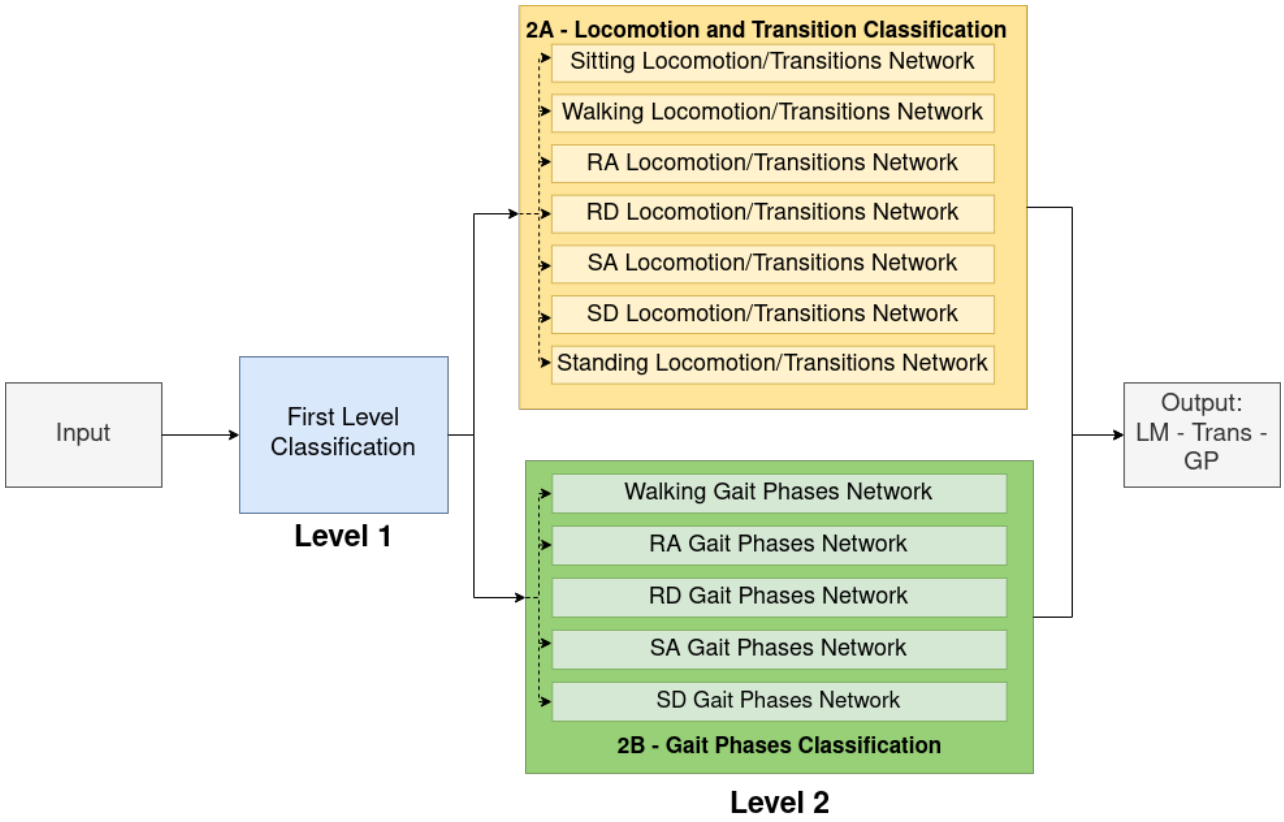



Figure 14: Proposed multi-level architecture.

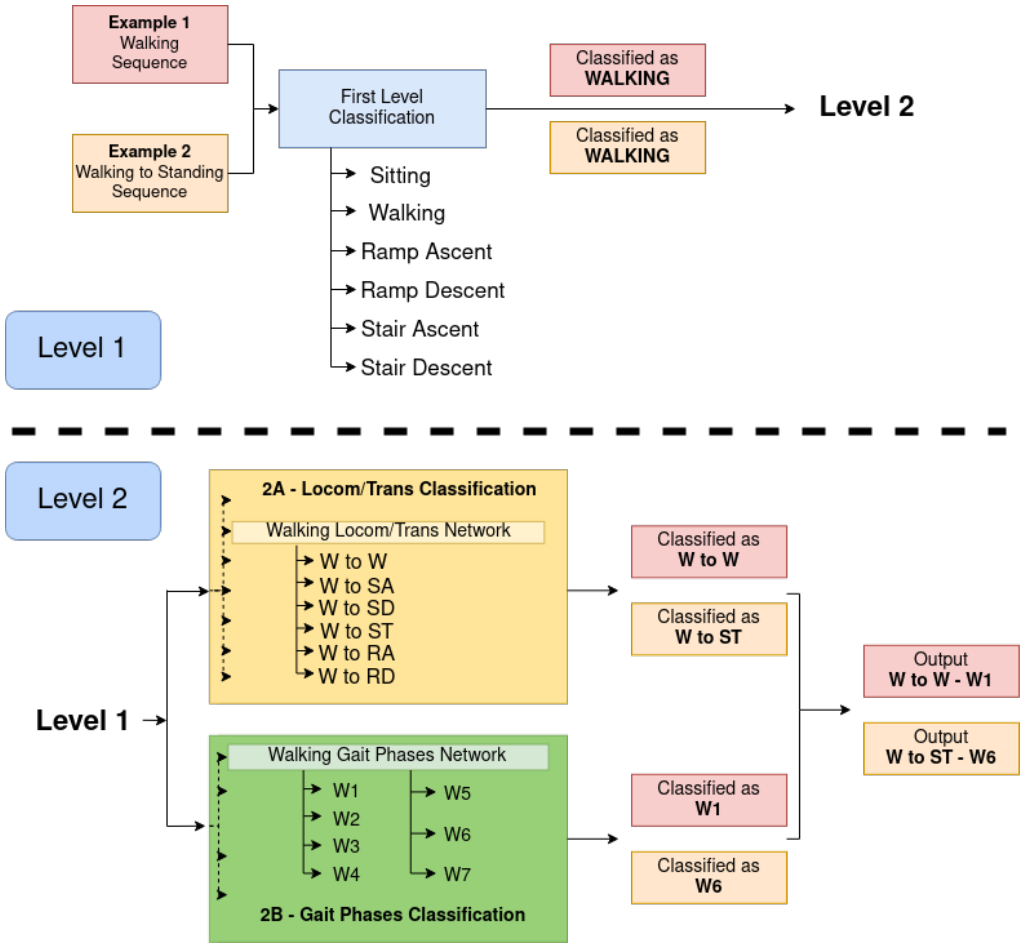



Figure 15: Example of classification of a given sequence.

## 4.2 Convolutional Neural Network

The architecture for the convolutional neural network is shown in figure 16. The input to the network is the mel-spectrogram image, which has size 10x50. The first two layers are convolutional layers with kernel size of (5x5) and number of filters equal to 64 and 128, respectively. They used a rectified linear unit and max-pooling of size (2x2). A dropout layer of value 0.25 follows. Finally, two dense layers of sizes 512 and 256, respectively, followed by a softmax layer whose size depends on the output of the network.

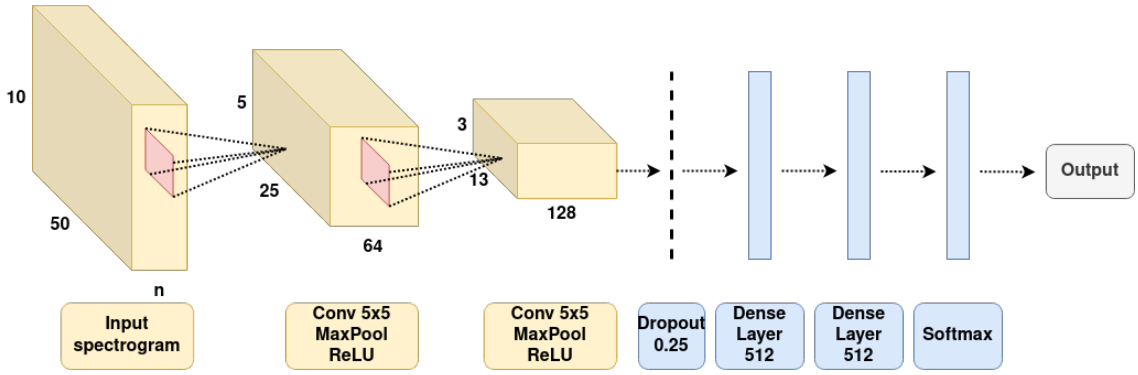



Figure 16: CNN architecture.

## 4.3 Convolutional Recurrent Neural Network

The architecture for the convolutional recurrent neural network can be seen in figure 17 for the CNN + GRU architecture and in figure 18 for the CNN + LSTM architecture. The input to the network is the mel-spectrogram image, which has size 10x50. The first two layers are convolutional layers with kernel size of (5x5) and number of filters equal to 64 and 128, respectively. Following there are recurrent layers, which are either two GRU layers or two LSTM layers with 120 and 60 units respectively. After that, there is a dense layer of size 30 followed by a dropout layer of value 0.25, To finish a softmax layer whose size depends on the output of the network is used.

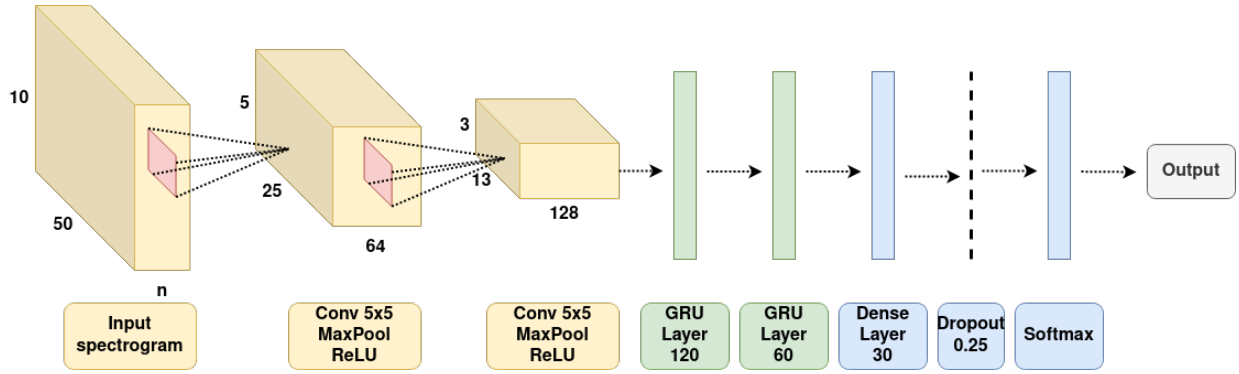



Figure 17: CNN + GRU architecture.

## 4.4 Evaluation: Performance Metric

As it can be seen in Tables 2, 3 and 4, there is some imbalance among the number of sequences when compared with other classes. Because of this, it was preferred to use the F1-score to measure

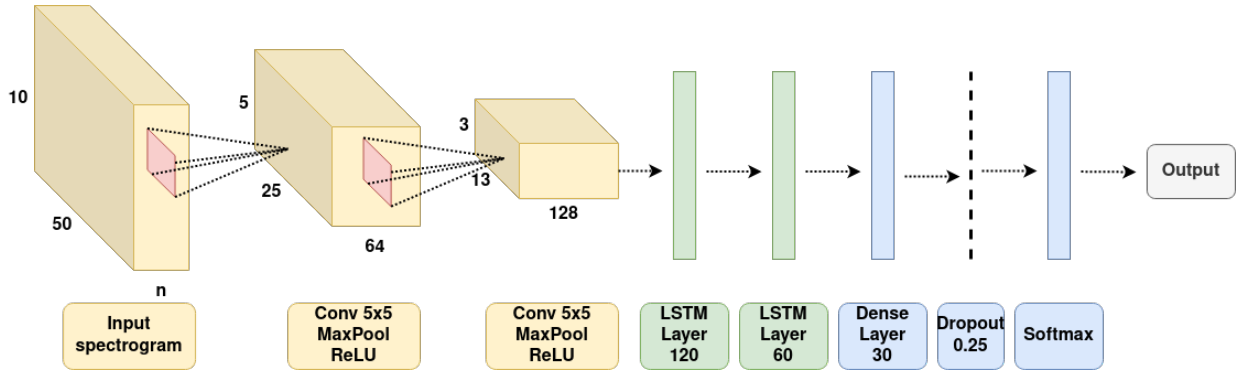



Figure 18: CNN + LSTM architecture.

the performance of the deep neural networks. It can be calculated as:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{3}$$

where $precision = tp/(tp+fp)$ and $recall = tp/(tp+fn)$, with $tp$ being the number of true positive predictions, $fp$ the number of false positives and $fn$ the number of false negatives.

## 4.5 Hyperparameters

This Section describes the main hyperparameters that were taken into consideration during the training process. The values for these hyperparameters are used as a baseline, since later on a grid search will be conducted on each individual network to optimize their performance. Still, for the initial analysis, base values are needed to train the network. The training is done on one computer with a GeForce RTX 2080 SUPER, AMD Ryzen 7 3700X 8-Core Processor and 8 GB RAM.

### 4.5.1 Learning Rate

The learning rate is set at 0.0001. A high value for the learning rate could cause the network to never converge while a very low learning rate could cause the network to get stuck in a local minima, thus the chosen value.

### 4.5.2 Optimizer

In this research the Adaptive Moment Estimation (Adam) optimizer has been used to optimize the gradient descent while training the network [33]. Adam computes individual adaptive learning rates

for different parameters.

### 4.5.3 Loss Function

For the loss function, it was decided to use the categorical cross-entropy.

### 4.5.4 Class Weighting

Since there is class imbalance in the data used for training as shown before,it si necesasry the implementation of a method to counteract this effect. One way is to use a class weight function like the one provided by the sklearn.utils module of the scikit-learn Python library [34]. This way the network penalizes more mistakes made for the underrepresented classes, in this case the transitions, without the need to create augmented data or vary the number of samples in the dataset.

### 4.5.5 Epochs

The data is presented 400 times to the networks during training to optimize data use. Even though it is a high number of epochs, early stopping will stop the training in advance if necessary.

### 4.5.6 Early Stopping

Given that there is a high number of epochs, the network has the risk of overfitting. To avoid it, early stopping is used. This technique will stop the training of the neural network if there has not been a sufficient improvement on the validation loss for 10 epochs. The minimum difference to consider an improvement in the validation loss is 0.001.

## 4.6 Experimental Setting

There are a number of steps taken in this project from the optimization of the neural networks to obtaining the final results for each dataset in different settings. All of it is explained in the following list:

- Train each deep neural network architecture both for one and two IMUs.

- Use the results from the previous step to find the best performing architecture by using a paired t-test.
- Optimize best performing network architecture with grid search.
- Train and test on the EN-ABL3S dataset on a subject dependent basis, meaning that information from the testing subject is included in the training process.
- Train and test on the EN-ABL3S dataset on a subject independent basis, meaning that the training process is independent from the testing subject as it does not contain related information.
- Train and test on the MyLeg dataset.
- Test effect of training with healthy subjects and testing on amputee subjects (En-ABL3S + MyLeg).
- Evaluate system prediction time.

# 5 Results

In this Section, the above mentioned steps are executed in order to obtain all the established results. The results are reported separately for each step and distinguishing between one and two IMUs. In the end, this results are discussed and compared to the current literature.

## 5.1 Individual Network Optimization

As shown in figure 14, there are many neural networks in the final system, each one in charge of classifying a certain subset of classes. In this step, all the different network architectures are taken into consideration for each one of the classification problems, both for one or two IMUs. In this step, networks are trained and tested on individual subjects from the EN-ABL3S dataset and the final F1-score is the result of averaging individual F1-scores.

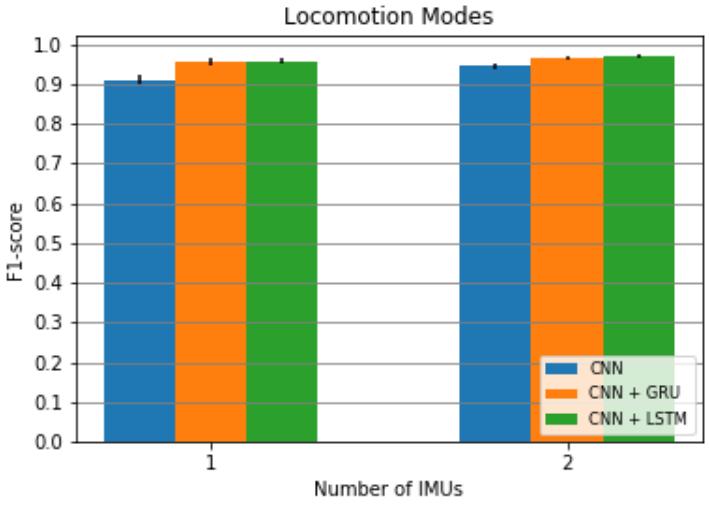
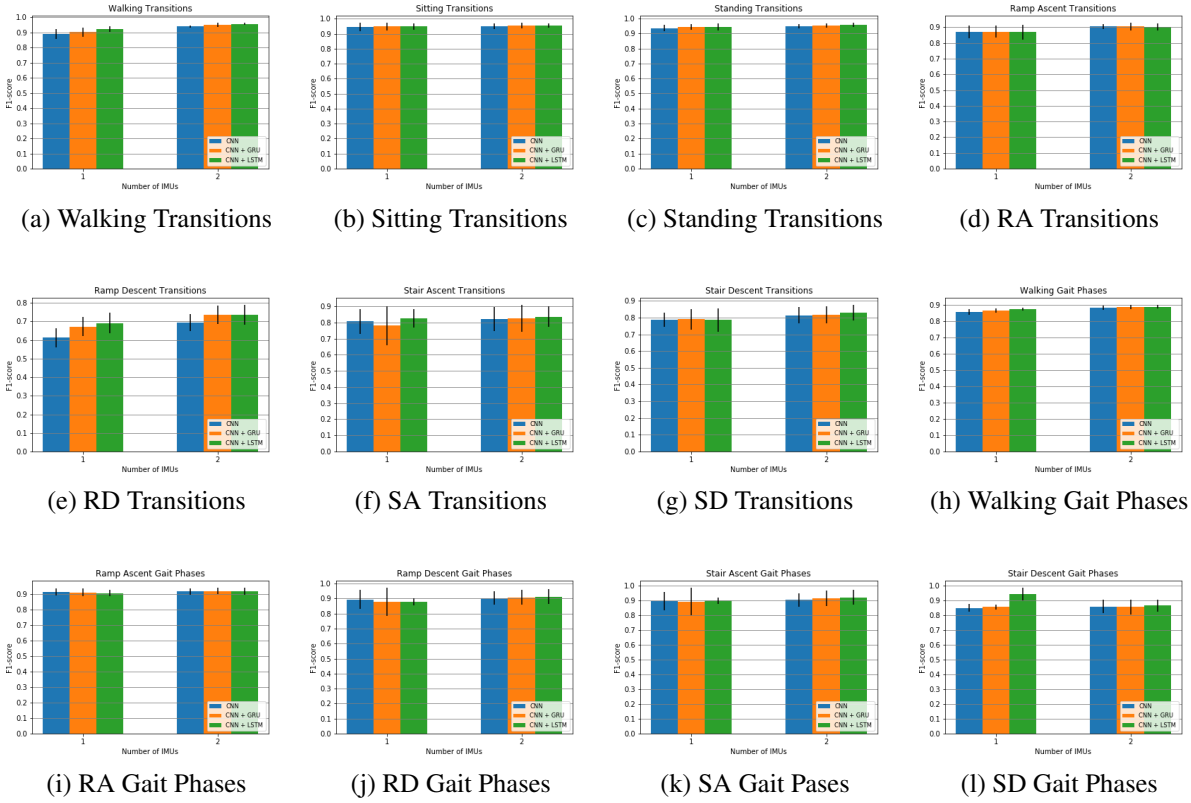



Figure 19: F1-score on different network architectures for one and two IMUs for the classification task of level 1.

As it can seen in Figures 19 and 20, the architecture corresponding to CNN + LSTM is obtaining the best F1-scores or is tied with the rest of the networks in most of the cases shown. Additionally, according to a paired t-test, there are significant differences between CNN + LSTM and the other architectures (CNN and CNN + GRU) in the "Locomotion Modes Classification" (figure 19) with p-values of $6.20 * 10^{-5}$ and $3.47 * 10^{-5}$ respectively and the "Walking Transitions" (figure 26a) with




(a) Walking Transitions (b) Sitting Transitions (c) Standing Transitions (d) RA Transitions

(e) RD Transitions (f) SA Transitions (g) SD Transitions (h) Walking Gait Phases

(i) RA Gait Phases (j) RD Gait Phases (k) SA Gait Pases (l) SD Gait Phases

Figure 20: F1-score on different network architectures for one and two IMUs for the classification tasks of level 2.

p-values of 0.001 and 0.021. For this reason, it was decided to use the CNN + LSTM architecture for the subsequent steps, both for the hyperparameter optimization and the final results.

For the hyperparameter optimization, grid search with 5 fold cross-validation is used. The best performing set of hyperparameters for each individual network obtained is used in future experiments. Among the hyperparameters considered for the optimization it is possible to find the activation function, the learning rate, optimizer, number of hidden units and the dropout value. Table 5 shows the results from the grid search and thus, the configuration each neural network will have during the remaining of the study.

.

| Classification | Activation | Learning Rate | Optimazer | Hidden Units | Dropout | Accuracy |
|---|---|---|---|---|---|---|
| locomModes | relu | 0.001 | RMSProp | [32, 64, 60, 30, 15] | 0.25 | 0.96 |
| sTrans | relu | 0.001 | Adam | [32, 64, 60, 30, 15] | 0.25 | 0.96 |
| wTrans | relu | 0.001 | Adam | [32, 64, 60, 30, 15] | 0.5 | 0.97 |
| saTrans | elu | 0.001 | RMSProp | [64, 128, 120, 60, 30] | 0.25 | 0.93 |
| sdTrans | elu | 0.001 | RMSProp | [64, 128, 120, 60, 30] | 0.25 | 0.92 |
| raTrans | elu | 0.001 | RMSProp | [64, 128, 120, 60, 30] | 0.25 | 0.96 |
| rdTrans | elu | 0.001 | RMSProp | [32, 64, 60, 30, 15] | 0.25 | 0.92 |
| stTrans | relu | 0.001 | Adam | [32, 64, 60, 30, 15] | 0.25 | 0.96 |
| wGaitPhases | tanh | 0.0001 | Adam | [64, 128, 120, 60, 30] | 0.5 | 0.92 |
| saGaitPhases | relu | 0.001 | RMSProp | [64, 128, 120, 60, 30] | 0.25 | 0.94 |
| sdGaitPhases | tanh | 0.001 | RMSProp | [32, 64, 60, 30, 15] | 0.25 | 0.92 |
| raGaitPhases | elu | 0.001 | Adam | [32, 64, 60, 30, 15] | 0.5 | 0.96 |
| rdGaitPhases | tanh | 0.001 | Adam | [64, 128, 120, 60, 30] | 0.25 | 0.95 |

Table 5: Grid search results. Best hyperparameters combination for each individual network.

## 5.2 EN-ABL3S Subject Dependent

This experiment consists on training the system on each healthy subject from the EN-ABL3S dataset, independently of each other, and test it on themselves. The objective is to evaluate the base performance of the networks on a personal subject, where all data used both for training and testing belongs to the same user. Table 6 shows the results obtained for this experiment. F1-score is obtained after averaging the individual F1-score of the ten subjects. The scenario in which two IMUs are used obtains the highest F1-scores with a value of $0.910 \pm 0.007$. Additionally, it obtains an F1-score of $0.926 \pm 0.007$ in the locomotion modes and transition classification (level 2A), and F1-score of $0.945 \pm 0.007$ in the gait phases classification (level 2B). The confusion matrixes for these two situations can be seen in Figures 21 and 21.

| F1-score 1 IMU | F1-score 2 IMUs |
|---|---|
| 0.893 ± 0.006 | 0.910 ± 0.007 |

Table 6: F1.score for CNN + LSTM configuration on a subject dependent scenario. Results obtained after averaging F1-score from 10 healthy subjects.

## 5.3 EN-ABL3S Subject Independent

This section tried to test the generalization capabilities of the system by testing on a novel subject. From the EN-ABL3S dataset, nine subjects are used for training and the remaining one is used for

testing. After that, we retrain the network with the data from the missing subject to see how much the network is able to improve. This process is repeated until all subjects had been used for testing once. Table 7 shows the results obtained for this experiment. In the scenario with all gait phases included, before retraining with the missing data the highest F1-score is obtained with two IMUs with a value of $0.612 \pm 0.045$. After retraining, the F1-score improves to $0.908 \pm 0.018$.

| | **F1-score 1 IMU** | **F1-score 2 IMUs** |
|---|---|---|
| **Before** | 0.495 ± 0.032 | 0.612 ± 0.045 |
| **After** | 0.892 ± 0.015 | 0.908 ± 0.018 |

Table 7: F1-score for CNN + LSTM configuration on a subject independent scenario. Results obtained after averaging the F1-score from 10 healthy subjects.

## 5.4 MyLeg Subject Dependent

In this experiment we test the system on a osseointegrated lower-limb amputee. Table 8 shows that using two IMUs obtains the best performance with an F1-score of $0.947 \pm 0.005$. In Figure 23, it can be seen the confusion matrix with the predictions after testing with the amputee subject.

| **F1-score 1 IMU** | **F1-score 2 IMUs** |
|---|---|
| 0.921 ± 0.006 | 0.947 ± 0.005 |

Table 8: F1.score for CNN + LSTM configuration on a subject dependent scenario. Results obtained from one osseointegrated lower-limb amputee.

## 5.5 MyLeg + EN-ABL3S Subject Independent

In this scenario, the performance of mixing healthy with amputee data is tested. Initially the system is trained with both one healthy subject or ten healthy subjects from the EN-ABL3S dataset and tested on the amputee data from the MyLeg dataset. After that, the system is retrained with the amputee information and tested again too see if there is any kind of improvement. At this point, the system is tested also on the healthy data to check the effect of retraining a the networks with amputee data.

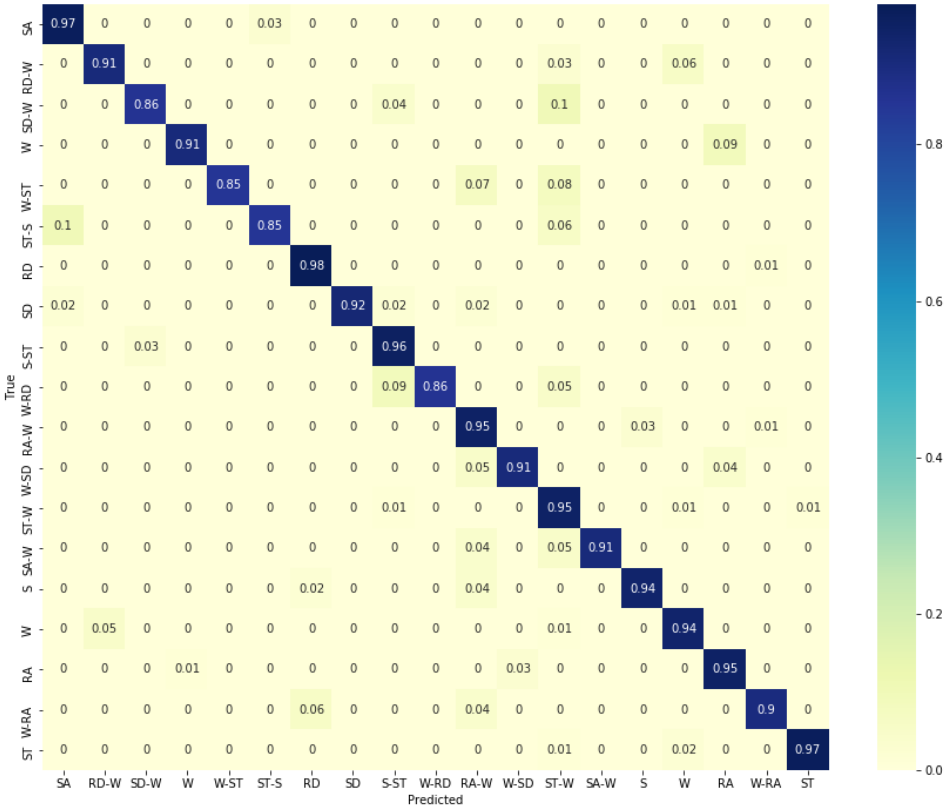



Figure 21: Confusion matrix for the EN-ABL3S subject dependent scenario on the CNN + LSTM configuration for the locomotion modes and transitions classification (level 2A).

As seen in Table 9, when using only one EN-ABL3S subject, initial peak F1-scores are $0.240 \pm 0.058$ and $0.213 \pm 0.043$ for one and two IMUs respectively, while after retraining with amputee data they go up to $0.873 \pm 0.009$ and $0.953 \pm 0.007$. In the case of using ten healthy subjects, as shown in Table 10, initial peak F1-scores are $0.288 \pm 0.013$ and $0.221 \pm 0.019$ for one and two IMUs respectively, while after retraining with amputee data they go up to $0.840 \pm 0.021$ and $0.946 \pm 0.005$.

Testing on the healthy subjects after retraining the system on the amputee data produces low performance, as seen in Tables 9 and 10, where peak F1-scores of $0.368 \pm 0.027$ and $0.335 \pm 0.008$ are obtained when using two IMUs for one and ten healthy subjects respectively.

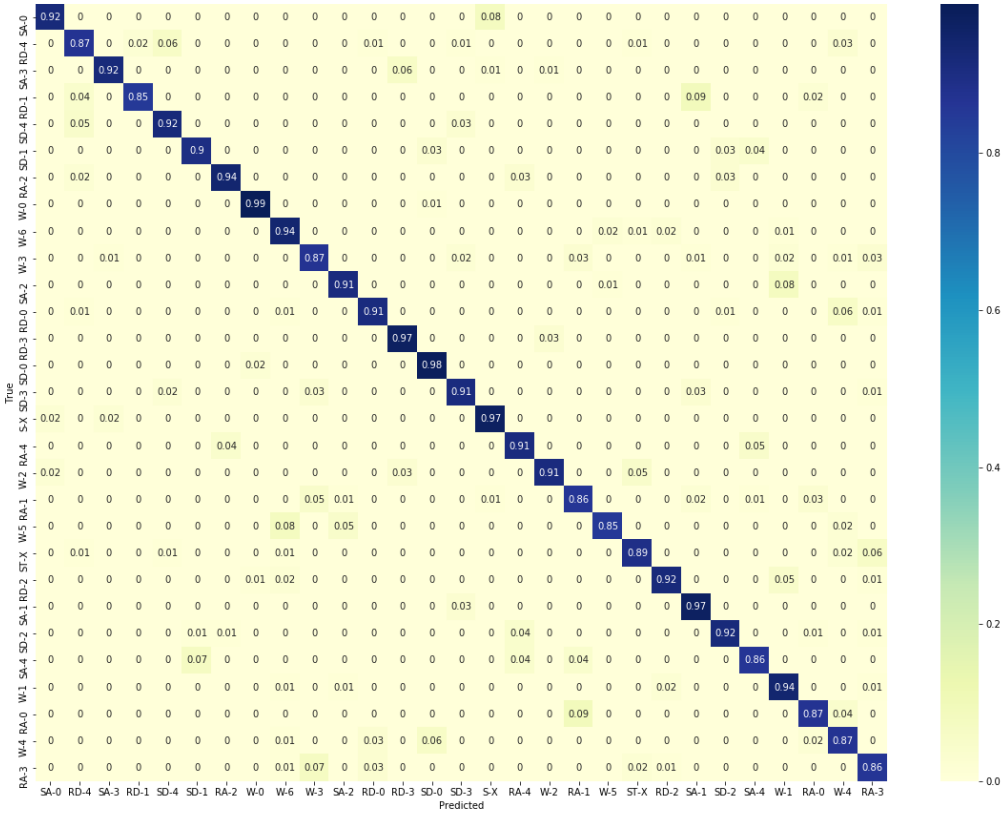



Figure 22: Confusion matrix for the EN-ABL3S subject dependent scenario on the CNN + LSTM configuration for the gait phases classification (level 2B).

## 5.6 Running Time

Table 11 shows the average running time for the classification of one sequence, averaged over 1000 sequences, each one containing six or twelve spectrograms (one or two IMUs respectively) extracted from 1.3 seconds of data. Both for one and two IMUs, the time necessary to perform the classification of one sample is under 50 ms, which is the sliding window time that is set to obtain a new sample, and is also below 300 ms, which was the maximum time allowed not to cause any discomfort to the user of the prosthesis.

It is worth noting that there is no clear difference between classifying samples from one IMU or

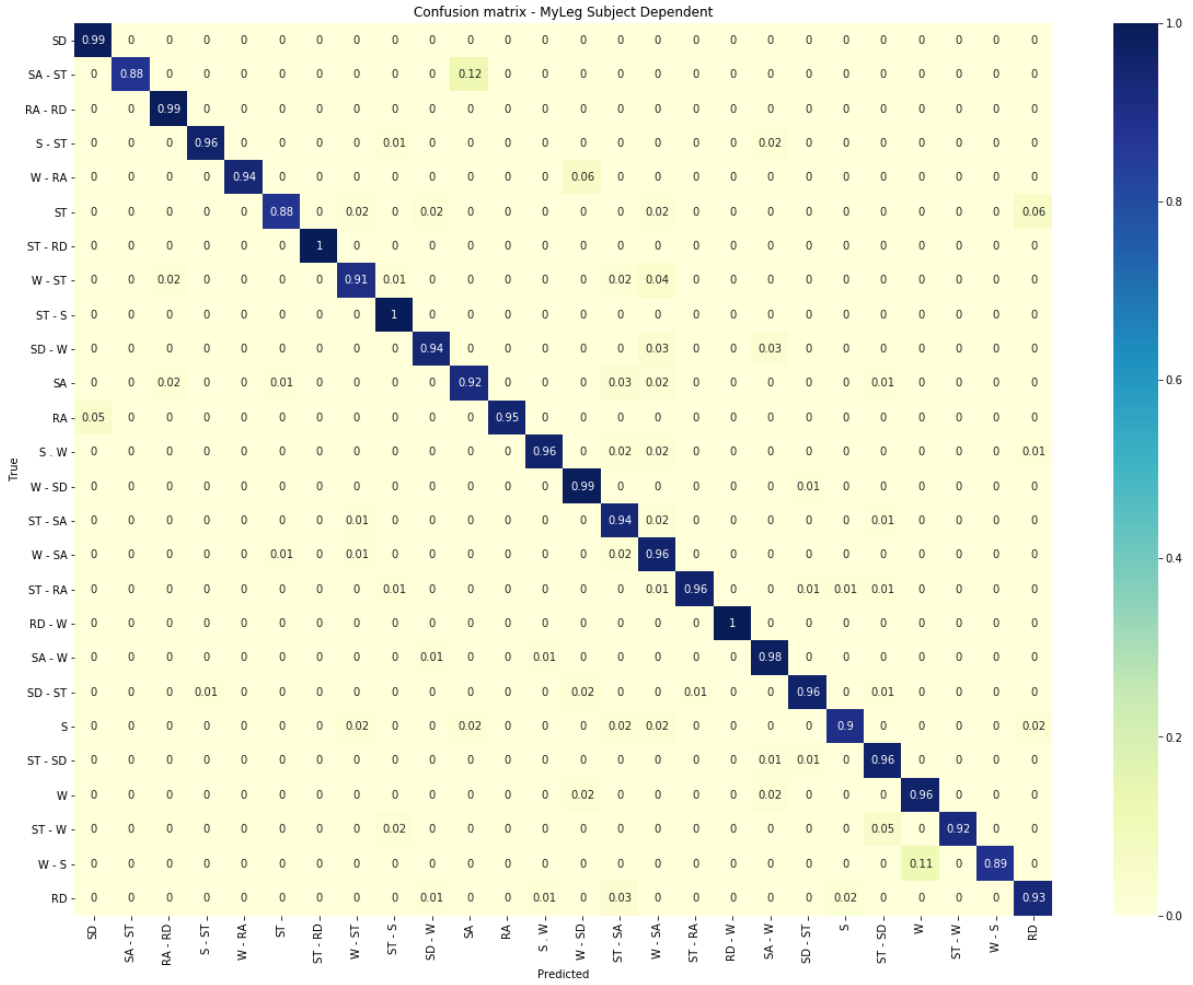



Figure 23: Confusion matrix for MyLeg subject dependent scenario on the CNN + LSTM configuration.

from two IMUs; there is only a difference in computation time when obtaining the spectrograms, since the amount of signals to process is double.

| | F1-score 1 IMU | F1-score 2 IMUs |
|---|---|---|
| **Before** | 0.240 ± 0.058 | 0.213 ± 0.043 |
| **After** | 0.873 ± 0.009 | 0.953 ± 0.007 |
| **EN-ABL3S** | 0.325 ± 0.023 | 0.368 ± 0.027 |

Table 9: 1 EN-ABL3S subject

| | F1-score 1 IMU | F1-score 2 IMUs |
|---|---|---|
| **Before** | 0.288 ± 0.013 | 0.221 ± 0.019 |
| **After** | 0.840 ± 0.021 | 0.946 ± 0.005 |
| **EN-ABL3S** | 0.239 ± 0.031 | 0.335 ± 0.008 |

Table 10: 10 EN-ABL3S subjects

| | Time 1 IMU [ms] | Time 2 IMUs [ms] |
|---|---|---|
| **Spectrogram** | 2.453 ± 0.021 | 4.785 ± 0.043 |
| **Classification** | 36.734 ± 5.612 | 37.091 ± 6.448 |
| **Total** | 39.187 ± 5.633 | 41.876 ± 6.491 |

Table 11: Running time for the prediction of one sample (averaged over a 1000 samples) when using one and two IMUs.

# 6 Discussion

## 6.1 EN-ABL3S Subject Dependent

In the case of the EN-ABL3S dataset a peak F1-score of $0.910 \pm 0.007$ when using two IMUs while an F1-score of $0.893 \pm 0.006$ for one IMU is obtained as shown in Table 6. Despite the overall good results obtained, taking a look at the confusion matrixes shown in figure 21 and 22, it can be seen some of the classes obtain an individual F1-score of around 0.85. This problem has to do mainly with transitions; either with transitions not being classified correctly or locomotion mode classes that are mistakenly taken as transitions. The reason for this behaviour could be that transitions classes are normally the under-represented classes which, despite the fact that we use class weighting when training the networks, they might still learn more information from the over-represented classes.

It must also be taken into account the effect of having multi-level architecture in the final overall F1-score. One wrong classification in the first level will undoubtedly produce wrong classifications in the subsequent levels, with the consequent drop in the final overall performance.

## 6.2 EN-ABL3S Subject Independent

In this experiment, the generalization ability of the system was put to the test. From the results in table 7 it can be seen the generalization requirements are not met, since only a peak F1-score of $0.612 \pm 0.045$ when using two IMUs is obtained. While it could be argued that this is a decent result considering that there is no information whatsoever from the novel subject in the system, training directly with the subject's data produces far better results as seen in table 6, where a peak F1-score of $0.910 \pm 0.007$ for two IMUs is obtained.

What is more, retraining with the missing subject data after training on the other nine subjects did not change the results significantly; from $0.893 \pm 0.006$ to $0.892 \pm 0.015$ in the case of one IMU and from $0.908 \pm 0.018$ to $0.910 \pm 0.007$ when using two IMUs.

It could be discussed that this lack of generalization might be given by the fact that different subjects move in different ways. Even though a locomotion mode across multiple subjects is essentially the same, differences in aspects such as the speed or the range of movement of the limbs might produce a reduction in the overall performance of the system.

## 6.3 MyLeg Subject Dependent

In the case of the MyLeg dataset, as shown in Table 8, a peak F1-score of $0.947 \pm 0.005$ is obtained when using two IMUs while an F1-score of $0.921 \pm 0.006$ for one IMU is obtained. Figure 23 shows the individual F1-scores for each of the classes from this dataset. As it happened with the EN-ABL3S dataset, some of the classes achieve an F1-score of around 0.85, which can be attributed to the class distribution. The main difference with respect to the EN-ABL3S dataset is that in this time there is no classification coming from level 2B as there are no gait phases involved, which makes the multi-level architecture be less influential in the overall F1-score.

## 6.4 EN-ABL3S + MyLeg Scenario

In this experiment, it was tested the influence that healthy subject data might have on the amputee data. As seen in Tables 9 and 10, testing the amputee data on a system trained with healthy data produces undesirable results (F1-score peaks at $0.288 \pm 0.013$ when using one IMU). This result can be understood by realizing that healthy and amputee movements are not comparable given the differences in locomotor capabilities between these two groups.

After retraining with the amputee data, F1-score reaches $0.953 \pm 0.007$ and $0.946 \pm 0.005$ for two IMUs when considering one and ten healthy subjects respectively, which is not a significant increment when compared to the F1-score obtained in table 8 ($0.947 \pm 0.005$). This also implies that there is no effect on pretraining the networks on a different number of subjects since the results will most likely be the same.

In fact, this process has made the networks to forget all information related to the healthy subjects as can be seen in row 3 of tables 9 and 10. Testing with healthy subjects after retraining with the amputee subject got a peak F1-score of $0.368 \pm 0.027$ and $0.335 \pm 0.008$ when using two IMUs for one and ten healthy subjects respectively.

## 6.5 Comparison to State-of-the-art

In this section, the results found in the literature, as shown in table 1, will be compared with the results obtained in this project.

Regarding previous works in which they were only classifying or detecting gait phases we find

that most of them focus on the major gait phases of the gait cycle such as swing and stance, or in the detection of gait events such as heel strike or toe off. In [13] they detect more in depth gait phases suck as loading response or push-off. These project obtain impressive results that sometimes reach 100% accuracy as in [11] and [12]. In comparison, in this project it was able able to keep an F1-score of 0.938 for healthy subjects while considering a wide variety of locomotion modes and transitions, with all their corresponding gait phases.

With respect to the locomotion mode and transition classification, most of the previous work compared comparably to the results obtained in this project. Special mention to [18], they use spectrograms to detect locomotion modes and transitions with an error rate of 1.1% but making use of up to 8 different sensors and no detection of gait phases whatsoever. In the case that they use only IMUs, their error rate falls to 4.68%. In the case of [23], they obtain an F1-score of 93.06 while here 94.7 is obtaining, and their proposed method does not take into account gait phases. Also, when using only 1 IMU, their results fall behind the ones obtained here: 92.1 against the 84.78.

Apart from that, this project has studied the generalization capabilities among healthy and amputee subjects, showing that there is poor generalization given the differences in locomotion gaits among this two groups, with little to no improvement when retraining a network with amputee data after initially training it with healthy data.

Lastly, taking a look at the locomotion modes, transitions and gait phase classification, the results obtained in this project can compare to the ones obtained by [24] and [25], not taking into account that they use a higher number of IMUs and pressure sensors, do not classify as many locomotion modes and transitions, and their gait phase classification is limited to the swing and stance phases of the movement.

## 6.6 Limitations and Future Outlook

### 6.6.1 Transition extraction and dataset structure

One of the main problems that appeared during this study is reflected in Figures 21, 22 and 23, which is that transitions are usually wrongly classified. This might be due to the way in which transitions are extracted, which makes some transition sequences contain information about previous or subsequent locomotion modes.

A more comprehensive study of the transition sequence extraction, together with the obtainment of transition information directly from the subjects, could help improve the final performance of the whole system as well.

### 6.6.2 Real-time Implementation

In order for the real user not to feel any kind of discomfort, predictions must be made approximately in the first 300 ms. With this approach, one sequence takes around 40 ms (around 4 ms to process the sample and 37 ms to classify the sequence), averaged over 1000 sequences, each one extracted from 1.3 seconds of data, which no only is inside the 300 ms range but also inside the 50 ms sliding window by which a new sequence would be ready to be processed.

On a real prosthesis, it must be taken into account that, for the experiments in this project, all necessary neural networks were loaded into memory at once since there was no need to worry about memory efficiency. On a real device this might not be possible and neural networks must need to be loaded every time they are needed and discarded right after, which might increase the amount of time necessary to process one sequence. On the other hand, in the experiments every classification was run sequentially (level 1, then level 2A, then level 2B), so there is room for improvement in the running time if some parallelism approaches are taken into consideration when designing the whole system.

### 6.6.3 Clinical Requirements

Future research should focus on the implementation and evaluation of the proposed method on osseointegrated amputees in clinical trials. This research could continue from the results obtained in the present since overall F1-score for amputee subjects reaches 0.947, plus individual classes are mostly all the of them over 0.9, as seen in figure 23. F1-scores might improve by the addition of new more complete data to the datasets since, as a general rule of thumb, in deep learning the more data you have the better the final performance will be.

# 7 Conclusions

This paper shows the research done in the design of a system for the real-time prediction of locomotion modes, transitions and gait phases for both healthy and osseointegrated lower-limb amputee subjects by using inertial measurement units (IMU). Different neural network configurations are investigated by combining convolutional and recurrent layers. As input to the networks, the frequency aspect in the form of an spectrogram, of one IMU ( located in the thigh) or two IMUs (located in both the thigh and the shank) are used. The system is able to predict 7 different locomotion modes (sitting, standing, walking, ramp ascent and descent, stair ascent and descent), transitions among this locomotion modes and the gait phases corresponding to each of the locomotion modes.

The results show that a system composed of CNN + LSTM networks is able to correctly predict user intention with a mean F1-score of $0.893 \pm 0.006$ and $0.910 \pm 0.007$ for the healthy subjects (considering all gait phases), and $0.921 \pm 0.006$ and $0.947 \pm 0.005$ for the amputee subject (only locomotion modes and transitions), using one and two IMUs respectively.

It has also shown that generalization capabilities for this type of classification task might be difficult to achieve given the nature of the data that is used, and that healthy and amputee data should not be mixed since it worsens the performance of the classification. It also sets some guidelines for future developments of similar projects.

# References


[1] B. Hudgins, P. Parker, and R. N. Scott. "A new strategy for multifunction myoelectric control". In: *IEEE Transactions on Biomedical Engineering* 40.1 (1993), pp. 82–94. DOI: 10.1109/10.204774.

[2] Binbin Su, Christian Smith, and Elena Gutierrez Farewik. "Gait Phase Recognition Using Deep Convolutional Neural Network with Inertial Measurement Units". In: *Biosensors* 10.9 (2020). ISSN: 2079-6374. DOI: 10.3390/bios10090109. URL: https://www.mdpi.com/2079-6374/10/9/109.

[3] A. J. Young and L. J. Hargrove. "A Classification Method for User-Independent Intent Recognition for Transfemoral Amputees Using Powered Lower Limb Prostheses". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 24.2 (2016), pp. 217–225. DOI: 10.1109/TNSRE.2015.2412461.

[4] Y. Lecun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Neural Information Processing Systems* 25 (Jan. 2012). DOI: 10.1145/3065386.

[6] Yann LeCun, Y. Bengio, and Geoffrey Hinton. "Deep Learning". In: *Nature* 521 (May 2015), pp. 436–44. DOI: 10.1038/nature14539.

[7] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. URL: https://doi.org/10.1162/neco.1997.9.8.1735.

[8] C. Olah. "Understanding LSTM Networks". In: (2015). URL: https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[9] Kyunghyun Cho et al. "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/D14-1179. URL: https://aclanthology.org/D14-1179.

[10] Jeff Donahue et al. *Long-term Recurrent Convolutional Networks for Visual Recognition and Description*. 2016. arXiv: 1411.4389 [cs.CV].

[11] E. D. Ledoux. "Inertial Sensing for Gait Event Detection and Transfemoral Prosthesis Control Strategy". In: *IEEE Transactions on Biomedical Engineering* 65.12 (2018), pp. 2704–2712. DOI: 10.1109/TBME.2018.2813999.

[12] H. F. Maqbool et al. "Real-time gait event detection for lower limb amputees using a single wearable sensor". In: *Proceedings of the IEEE International Conference Engineering in Medicine*. 2016, pp. 5067–5070. DOI: 10.1109/EMBC.2016.7591866.

[13] Binbin Su, Christian Smith, and Elena Gutierrez Farewik. "Gait Phase Recognition Using Deep Convolutional Neural Network with Inertial Measurement Units". In: *Biosensors* 10.9 (2020). ISSN: 2079-6374. DOI: 10.3390/bios10090109.

[14] Yu Lou et al. "IMU-Based Gait Phase Recognition for Stroke Survivors: Preliminary Results". In: *IEEE International Conference on CYBER Technology in Automation, Control, and Intelligent Systems*. 2018, pp. 802–806. DOI: 10.1109/CYBER.2018.8688103.

[15] Wei-Han Chen et al. "Determining motions with an IMU during level walking and slope and stair walking". In: *Journal of Sports Sciences* 38 (Oct. 2019), pp. 1–8. DOI: 10.1080/02640414.2019.1680083.

[16] Rafael Rego Drumond et al. "PEEK - An LSTM Recurrent Network for Motion Classification from Sparse Data". In: *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications* (2018), pp. 215–222. DOI: 10.5220/0006585202150222.

[17] J. Figueiredo et al. "Daily Locomotion Recognition and Prediction: A Kinematic Data-Based Machine Learning Approach". In: *IEEE Access* 8 (2020), pp. 33.250–33.262. DOI: 10.1109/ACCESS.2020.2971552.

[18] U. H. Lee et al. "Image Transformation and CNNs: A Strategy for Encoding Human Locomotor Intent for Autonomous Wearable Robots". In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 5440–5447. DOI: 10.1109/LRA.2020.3007455.

[19] Benyue Su et al. "A CNN-Based Method for Intent Recognition Using Inertial Measurement Units and Intelligent Lower Limb Prosthesis". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 27 (2019), pp. 1032–1042. DOI: 10.1109/TNSRE.2019.2909585.

[20] Enhao Zheng and Qining Wang. "Noncontact Capacitive Sensing-Based Locomotion Transition Recognition for Amputees With Robotic Transtibial Prostheses". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25.2 (2017), pp. 161–170. DOI: 10.1109/TNSRE.2016.2529581.

[21] Zuojun Liu et al. "Intent pattern recognition of lower-limb motion based on mechanical sensors". In: *IEEE/CAA Journal of Automatica Sinica* 4.4 (2017), pp. 651–660. DOI: 10.1109/JAS.2017.7510619.

[22] Huaitian Lu, Lambert R B Schomaker, and Raffaella Carloni. "IMU-based Deep Neural Networks for Locomotor Intention Prediction". In: *Proceedings of the IEEE.RSJ International Conference on Intelligent Robots and Systems* (2020), pp. 4134–4139. DOI: 10.1109/IROS45743.2020.9341649.

[23] Julian Bruinsma and Raffaella Carloni. "IMU-Based Deep Neural Networks: Prediction of Locomotor and Transition Intentions of an Osseointegrated Transfemoral Amputee". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 29 (2021), pp. 1079–1088. DOI: 10.1109/TNSRE.2021.3086843.

[24] Dongfang Xu et al. "Real-Time On-Board Recognition of Continuous Locomotion Modes for Amputees With Robotic Transtibial Prostheses". In: *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 26.10 (2018), pp. 2015–2025. DOI: 10.1109/TNSRE.2018.2870152.

[25] Baojun Chen, Enhao Zheng, and Qi Wang. "A Locomotion Intent Prediction System Based on Multi-Sensor Fusion". In: *Sensors* 14 (2014). DOI: 10.3390/s140712349.

[26] Blair Hu, Elliott Rouse, and Levi Hargrove. "Benchmark Datasets for Bilateral Lower-Limb Neuromechanical Signals from Wearable Sensors during Unassisted Locomotion in Able-Bodied Individuals". In: *Frontiers in Robotics and AI* 5 (2018). ISSN: 2296-9144. DOI: 10.3389/frobt.2018.00014.

[27] U Themes. "Gait and Posture Analysis". In: (2019). URL: https://musculoskeletalkey.com/gait-and-posture-analysis.

[28] Xenia Karekla and Nick Tyler. "Maintaining balance on a moving bus: The importance of three-peak steps whilst climbing stairs". In: *Transportation Research Part A: Policy and Practice* 116 (2018), pp. 339–349. DOI: 10.1016/j.tra.2018.06.020.

[29] A. McIntosh et al. "Gait dynamics on an inclined walkway." In: *Journal of biomechanics* 39 (2006), pp. 2491–502.

[30] Amy Silder, Thor Besier, and Scott Delp. “Predicting the Metabolic Cost of Incline Walking from Muscle Activity and Walking Mechanics”. In: *Journal of biomechanics* 45 (2012), pp. 1842–9. DOI: 10.1016/j.jbiomech.2012.03.032.

[31] Douglas O’Shaughnessy. *Speech communication: human and machine*. Addison-Wesley, 1987, p. 150.

[32] Brian McFee et al. “librosa: Audio and Music Signal Analysis in Python”. In: 2015, pp. 18–24. DOI: 10.25080/Majora-7b98e3ed-003.

[33] Diederik Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *International Conference on Learning Representations* (2014).

[34] Fabian Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12.85 (2011), pp. 2825–2830.

# Appendix A Re-labelling of Original Data

This section covers more in depth aspects of the process of re-labelling the original data to create the sequences that later will be used in the training and testing of the neural networks.

The original data, initially, only contained labels for the locomotion modes. In spite of this, it contain additional information, such as toe off and heel strike events timestamps, that helped when re-labelling the data in order to also contain labels for the transitions and the gait phases.

## A.1 Transition Re-labelling

For transition re-labelling, the change point in between two different locomotion modes is used. The previous and the subsequent 250 ms (125 samples) to the change point are re-labelled as the transition between the two locomotion modes. Figure 24 shows a depiction of the re-labelling process.

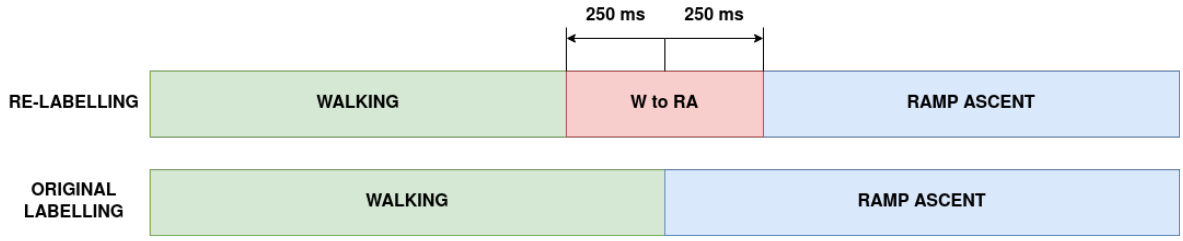



Figure 24: Transition re-labelling process.

## A.2 Gait Phases Re-labelling

For labelling the gait phases, the heel strike and toe off events information is used. In the original dataset, the timestamps of each of these events are provided so it is possible to delimit the swing and stance phases of the corresponding leg and label them accordingly. In between a heel strike and a toe off event there is the stance gait phase, while in between a toe off and a heel strike event there is the swing gait phases. After this, each gait phase (swing or stance) is further divided into extra gait phases according to the intervals shown in Figure 11. Figure 25 shows a depiction of the gait phases relabelling process.

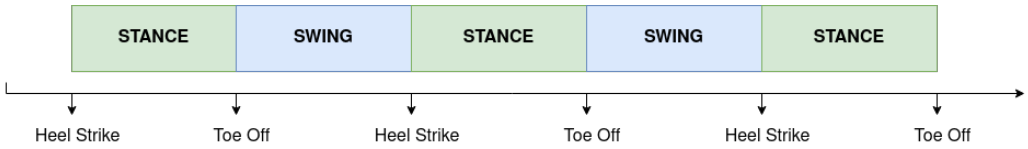



Figure 25: Gait Phases re-labelling process.

# Appendix B Spectrogram Algorithm

In this appendix it is explain in depth the process of computing the spectrogram of a given sequence as well as some code snippets are shown to illustrate this process. The code makes use of the library librosa for all the computations related with the spectrogram.

Listing 1 shows the process of computing the spectrogram of a given time signal. This time signal can correspond to either signal coming from the accelerometer or gyroscope in the IMUs. Initially, the mel spectrogram is computed in term of some parameters such as the sampling rate, the length of the FFT window or the window type. After that, the spectrogram is converted into decibels and later on is normalized in the range [0, 1] so it can be processed by the neural networks.

```
1  def computeStectrogram(data):
2
3      data = data.to_numpy()
4
5      S = librosa.feature.melspectrogram(y = data, sr = 500, n_fft=20,
6                                         hop_length=13, n_mels = 10,
7                                         window='hann', power=2, fmax=350)
8
9      S_dB = librosa.power_to_db(S, ref = np.max)
10     melSpec = np.flip(np.flip(S_dB), axis = 1)
11     melSpec = (melSpec - np.min(melSpec))/np.ptp(melSpec)
12
13     return melSpec
```

Listing 1: Spectrogram calculation for a time signal.

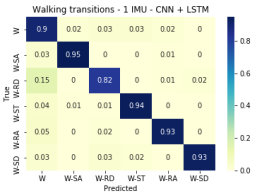
Listing 2 shows the function to obtain all the spectrograms necessary from a sequence containing all singlas from the IMUs. For each signal, the previously explained method **computeSpectrogram()** is used to extract said spectrogram. After that, all extracted spectrograms are concatenated together and returned to be used as input to the neural network.

```
def getSpectrogram(sequence):

    processedSequence = []

    # Number of col is 6 or 12 depending on number of IMUs
    for col in sequence:
        processedSequence.append(computeStectrogram(sequence[col]))

    return np.asarray(processedSequence)
```

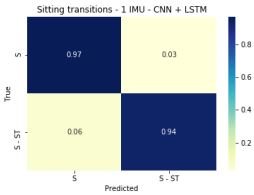
Listing 2: Obtain the spectrogram for every signal contained in a sequence.

# Appendix C Confusion Matrices for Individual Network Optimization

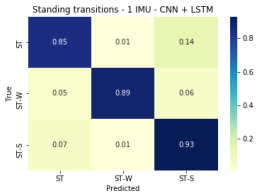
In this appendix it can be seen multiple figures showing the confusion matrixes for the performance of individual networks after the initial training, prior to the grid search optimization, as explained in Section 5.1.



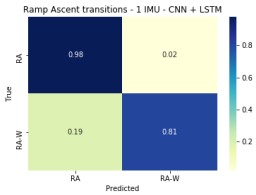

(a) Walking Transitions



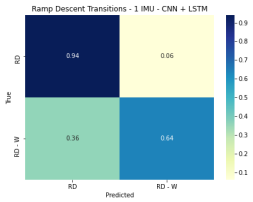

(b) Sitting Transitions



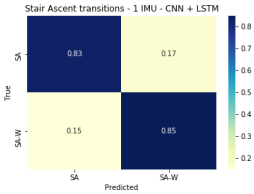

(c) Standing Transitions



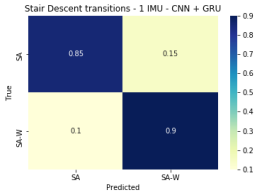

(d) RA Transitions



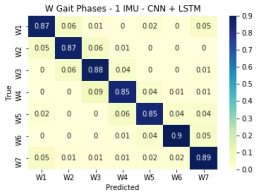

(e) RD Transitions



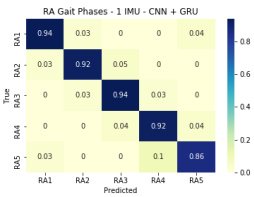

(f) SA Transitions



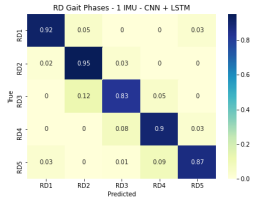

(g) SD Transitions



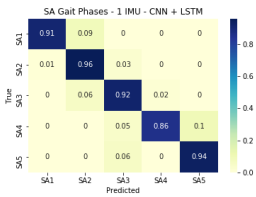

(h) Walking Gait Phases



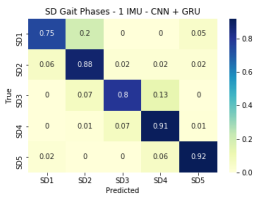

(i) RA Gait Phases




(j) RD Gait Phases




(k) SA Gait Pases




(l) SD Gait Phases

Figure 26: Confusion matrix for the best performing setup on the individual network optimization according to results in Figure 20.