Universidade do Minho
Escola de Engenharia

Diogo Barata Gonçalves

Towards real-time recognition and prediction of human and humanoid robot locomotion modes

Master's Dissertation
Master's Degree in Industrial Electronics and Computer Engineering

Dissertation supervised by
Professor Cristina Santos
Engineer Joana Figueiredo

January 26, 2018

DECLARAÇÃO

Nome: Diogo Barata Gonçalves
Endereço eletrónico: diobgo@gmail.com

Título da dissertação: "Towards real-time recognition and prediction of human and humanoid robot locomotion modes"

Orientadores:
Professora Doutora Cristina Santos
Ano de conclusão: 2018

Mestrado integrado em Engenharia Eletrónica Industrial de Computadores
Área de Especialização: Automação, Controlo e Robótica

É AUTORIZADA A REPRODUÇÃO INTEGRAL DESTA DISSERTAÇÃO APENAS PARA EFEITOS DE INVESTIGAÇÃO, MEDIANTE DECLARAÇÃO ESCRITA DO INTERESSADO, QUE A TAL SE COMPROMETE.

Universidade do Minho, _____/_____/_____

Assinatura: _____

ACKNOWLEDGEMENTS

All the work carried out during the past year wouldn't have been possible without the help and support from many people. I would to dedicate this section to the ones that most contributed, directly or indirectly, but to all those I don't mention here I am still very thankful for every bit of help given to me.

I would first like to thank Professor Cristina Santos for all the support and help as well as the opportunity to do this work and other opportunities I was provided with.

A huge thank you to Joana for all the guidance, constant support and patience even when busy to help me thread subjects I wasn't familiar with and constantly pushing me to do my best. Your diligence and work ethic are an example that I will take with me for my future life.

Thank you Intruja for designing the InertialLab, and for your support and incessant will to help. You were a good friend and I really appreciate it.

I am thankful to my friend David for his help on some of the tougher problems I faced and for keeping me on the right path.

I am also very thankful to all my laboratory colleagues for always making the lab such a wonderful environment to work in and keeping things light and fun.

Last but not least, I would like to thank my family for their continued support during the entirety of the time that took me to finish this work.

ABSTRACT

Several afflictions can affect a person's ability to walk from muscular impairments, weakness or neurologic injury. In many of these cases, rehabilitation is essential for a full recovery. With the advances in the field of robotics and its bigger integration in rehabilitation, namely in the form of active orthosis and prosthesis, novel solutions to old challenges are made available. One of these challenges is the ability to use these assistive devices seamlessly without expert intervention in a subject's daily life.

Faced with this hindrance it becomes important to develop strategies that can recognize and predict human locomotion modes to allow a timely and correct response to a user's needs from assistive devices. As such, this thesis proposes a pipeline of which the output is either a machine learning model that can recognize in real-time the user's current locomotion mode or one that can predict a user's walking intention.

The locomotion mode recognition model can identify walking direction (forward, backward, anti-clockwise and clockwise) as well as locomotion activities (level walking, stair ascent, stair descent, ramp ascent and ramp descent) in real-time. Similarly, the intention prediction model also predicts both direction and locomotion activity intention in a timeframe that allows an assistive device to preemptively act in a seamless manner to provide the user a fluid walking ability and avoid a fall due to improper terrain traversing manner.

An assessment of the required biomechanical features is done to identify the ones that best help predict or recognize the locomotion mode using feature selection methods (Principal Component Analysis, Analysis of variance-based selection, forward and backwards sequential selection).

Several classification algorithms (Support Vector Machines, K-nearest neighbors, random forests and discriminant analysis) were explored and implemented to find the best performing one. These models were tested with data from healthy human subjects and a humanoid robot with a human-like gait controller.

Results revealed that during the model building procedure using the Support Vector Machines algorithm with a feature selection method that combined the mRMR (minimum redundancy Maximum Relevancy) ranking technique and the forward feature selection procedure yielded the most robust and best-performing model. Direction prediction and recognition models presented an MCC (Matthews Correlation Coefficient) value of 0.98, on average, after validation showing promising results. However, and despite steady-state step

type models (models classifying non-transitional steps) having an MCC value of 0.98, models involved in the classification of transitional steps, both for recognition and prediction, revealed poor results. MCC values as low as 0.61 were reported, showing that the used features were inadequate for the prediction of a subject's gait intention.

Future work will be to integrate other kinds of sensors and use different features that can rectify the classification flaws present in the obtained models in order to increase their accuracy.

KEYWORDS: MACHINE LEARNING, HUMAN GAIT, FEATURE SELECTION, MOTION PREDICTION, MOTION RECOGNITION

RESUMO

Vários fatores podem afetar a capacidade de locomoção de uma pessoa desde lesões ou fraqueza musculares a lesões neurológicas. Em muito destes casos, reabilitação é essencial para uma completa recuperação. Com os avanços no campo da robótica e a sua maior integração em reabilitação, nomeadamente na forma de próteses e ortóteses ativas, novas soluções para velhos problemas tornam-se disponíveis. Um destes desafios é a habilidade de usar estes dispositivos assistivos de forma fluida e não-obstrutiva durante o dia-a-dia sem necessidade da intervenção de um especialista.

Face a este problema torna-se importante desenvolver estratégias que possibilitem o reconhecimento e previsão de modos de locomoção humanos para permitir uma resposta correta e pontual de dispositivos assistivos face ás necessidades do utilizador. Como tal, esta tese propõe uma *pipeline* que tem como resultado um modelo de *machine learning* que consegue reconhecer em tempo-real o modo de locomoção enquanto acontece ou um modelo que consegue identificar a intenção de locomoção do utilizador.

A *pipeline* delineada nesta tese permite obter um modelo que reconhece a direção de locomoção (frente, trás, anti-horário, horário) assim como a atividade locomotora (andar em terreno plano, subir escadas, descer escadas, subir rampas e descer rampas) em tempo real. A previsão de intenção também prevê tanto a direção como a atividade locomotora numa janela de tempo que permite ao dispositivo assistivo atuar sobre essa intenção.

Um estudo das características biomecânicas necessárias é feito para identificar aquelas que melhor ajudam na previsão ou reconhecimento do modo de locomoção usando métodos de *feature selection* (*Principal Component Analysis*, *ANOVA-based selection*, *forward* e *backwards sequential selection*).

Vários algoritmos de classificação (*Support Vector Machines*, *K-nearest neighbors*, *Random Forests* e *Discriminant Analysis*) foram explorados e implementados de forma a descobrir qual o melhor. Estes modelos foram testados com dados de sujeitos saudáveis e de um robô humanoide com um controlador de marcha humano.

Resultados revelaram que para, a construção do modelo, o uso do algoritmo SVM e a seleção de *features* através da combinação dos métodos mRMR e *forward feature selection* resultavam no melhor e mais robusto modelo. Os modelos de classificação de direção, tato de reconhecimento como de previsão, obtiveram um valor de MCC de 0.98 em média depois da validação mostrando-se promissores. No entanto, e apesar dos modelos de reconhecimento e

previsão de passos *steady-state*, terem obtido valores de MCC de 0.98, os modelos envolvidos na classificação de passos transicionais, tanto de reconhecimento como de previsão, obtiveram resultados fracos com valores de MCC tão baixos como 0.61, revelando que as *features* usadas são inadequadas para a previsão da intenção de marcha.

Como trabalho futuro deverão ser integrados outros tios de sensores e usadas outras *features* que possam retificar as falhas de classificação presentes os modelos obtidos de forma a aumentar a sua *perfromace*.

Palavras-Chave: Machine learning, Marcha Humana, Feature Selection, Previsão de movimento, reconhecimento de movimento

INDEX

Acknowledgements iii
Abstract v
Resumo vii
List of figures xiii
List of tables xv
Abbreviations and Acronyms xvii
1. Introduction 1
1.1 Motivation 1
1.2 Problem Statement 2
1.3 Goals and Research Questions 3
1.4 Thesis Outline 4
2. Theoretical Background 7
2.1 Assistive Devices 7
2.2 Machine Learning 8
2.3 Feature Engineering 9
2.4 Data Pre-processing 10
2.5 Data Normalization 10
2.6 Dimensionality Reduction 11
2.7 Classification Model Building 16
2.8 Model Validation 19
2.8.1 Validation methods 19
2.8.2 Performance metrics 20
3. State of the art 25
4. Methodology 45
4.1 Data Acquisition 46
4.1.1 Humanoid Robot 46
4.1.2 Human 47
4.2 Feature Calculation 50
4.3 Pre-processing 51
4.4 Data Labeling 52

4.5 Model Building 53
4.6 Model Evaluation 55
4.7 Online Classification 56
5. Procedure Tests 59
5.1 Feature Calculation 59
5.1.1 Methods 59
5.1.2 Results 60
5.1.3 Discussion 61
5.2 Normalization 61
5.2.1 Methods 61
5.2.2 Results 62
5.2.3 Discussion 66
5.3 Feature Selection and Extraction 67
5.3.1 Methods 67
5.3.2 Results 67
5.3.3 Discussion 68
5.4 Classifier Algorithm Selection 68
5.4.1 Methods 69
5.4.2 Results 69
5.4.3 Discussion 70
6. Locomotion Mode Recognition 73
6.1 Methods 73
6.2 Results 73
6.2.1 Humanoid Robot 74
6.2.2 Human 75
6.3 Discussion 77
7. Locomotion Mode Prediction 79
7.1 Methods 79
7.2 Results 79
7.2.1 Humanoid Robot 80

7.2.2 Human 81
7.3 Discussion 83
8. Conclusions 85
8.1 Final Considerations 85
8.2 Future work 87
References 89
Appendix I 94

LIST OF FIGURES

Figure 1 - Active orthosis 7
Figure 2 - Prostheis 8
Figure 3 - KNN-based classification example 17
Figure 4 - SVM Hyperplane example (the axes represent the feature values and each shape is a different class) 18
Figure 5 - Example of a confusion matrix 21
Figure 6 - Meaning of each designation in relation to class label P 21
Figure 7 - Pipeline Diagram 45
Figure 8 - DarwIn-Op in the simulated environment 47
Figure 9 - DarwIn-OP and the obtained joint angles (the red lines indicate the rotation axis) 47
Figure 10 - IMU placement 48
Figure 11 - Photos of the circuits where the trials were done 49
Figure 12 - Ongoing trials 50
Figure 13 - Schematic of classification models sequence for robot data 56
Figure 14 - Schematic of classification models sequence for human data 57
Figure 15 – Average Recognition Model Performance 60
Figure 16 – Average Prediction Model Performance 60
Figure 17 - Average MCC value of the classification model per normalization technique 62
Figure 18 - Average MCC value of the Linear DA models by normalization type 63
Figure 19 - Average MCC value of the Quadratic DA models by normalization type 63
Figure 20 - Average MCC value of the Regular KNN models by normalization type 63
Figure 21 - Average MCC value of the Weighted KNN models by normalization type 64
Figure 22 - Average MCC value of the Random Forests models by normalization type 64
Figure 23 - Average MCC value of the Linear SVM models by normalization type 65
Figure 24 - Average MCC value of the Quadratic SVM models by normalization type 65
Figure 25 - Average Performance of the Cubic SVM models by normalization type 65
Figure 26 - Average MCC value of the Gaussian SVM models by normalization type 66
Figure 27 - Average Performance per feature selection technique 67
Figure 28 - Number of selected features by feature selection technique 68

Figure 29 - Average MCC value of the models for each classification algorithm..... 70
Figure 30 - Average classification time for each classification algorithm..... 70

LIST OF TABLES

Table 1 – Results of the evaluation of the robot gait models 74
Table 2 - Confusion matrix of recognition model trained with database ‘direction_ft’ 74
Table 3 - Confusion matrix of recognition model trained with database ‘steady_state_type_ft’ 74
Table 4 - Selected features common to all robot gait recognition models 74
Table 5 - Results of the evaluation of the human gait classification models 75
Table 6 - Confusion matrix of the recognition model trained with database ‘direction_ft’..... 75
Table 7 - Confusion matrix of the recognition model trained with database ‘sts_trs_ft’..... 75
Table 8 - Confusion matrix of the recognition model trained with database ‘transition_type_ft’ (LW – level-walking; AS – ascending stairs; DS – descending stairs; AR – ascending ramps; DR – descending ramps; SO – stepping over obstacle) 76
Table 9 - Confusion matrix of the recognition model trained with database ‘steady_state_type_ft’ 76
Table 10 - Selected features common to all human gait recognition models 76
Table 11 – Results of the evaluation of the robot gait classification models 80
Table 12 - Confusion matrix of the recognition model trained with database ‘direction_ft’... 80
Table 13 - Confusion matrix of recognition model trained with database ‘steady_state_type_ft’ 80
Table 14 - Selected features common to all robot gait prediction models 80
Table 15 – Results of the evaluation of the human gait classification models 81
Table 16 - Confusion matrix of the recognition model trained with database ‘direction_ft’... 81
Table 17 - Confusion matrix of the recognition model trained with database ‘sts_trs_ft’..... 81
Table 18 - Confusion matrix of the recognition model trained with database ‘transition_type_ft’ (LW – level-walking; AS – ascending stairs; DS – descending stairs; AR – ascending ramps; DR – descending ramps; SO – stepping over obstacle) 81
Table 19 - Confusion matrix of the recognition model trained with database ‘steady_state_type_ft’ 82
Table 20 - Selected features common to all human gait prediction models 82

ABBREVIATIONS AND ACRONYMS

ANOVA – Analysis of Variance

AUC – Area Under Curve

CPG – Central Pattern Generator

CV – Cross-Validation

DA – Discriminant analysis

DOF – Degrees of Freedom

EMG – Electromyography

FN – False Negatives

FP – False Positives

HS – Heel Strike

IMU – Inertial Measurement Unit

KNN – K-nearest Neighbors

kPCA – Kernel Principal Component Analysis

LDA – Linear Discriminant Analysis

LOO – Leave-One-Out

MCC – Matthews Correlation Coefficient

mRMR – Minimum Redundancy Maximum Relevancy

PA – Parallel Analysis

PCA – Principal Component Analysis

RF – Random Forests

SVM – Support Vector Machines

TP – True Positives

TN – True Negatives

TO – Toe-Off

1. INTRODUCTION

The work detailed in this document was developed on the area of rehabilitation during the past year in the Adaptive System Behaviour Group (ASBG) of the CMEMS from University of Minho.

This work addresses the field of human gait recognition and prediction and the goal of this dissertation is to develop machine learning models that can when applied to lower-limb assistive devices and in real-time, accurately assess the user's current locomotion mode and predict future ones in order to help the device transition between control modes.

This chapter introduces and contextualizes the work done and the challenges being tackled. It also summarizes the goals and outline of the document as well.

1.1 Motivation

Human walking is a complex repetitive task that depends on muscular strength, joint mobility and coordination of the central nervous system (Plotnik, Bartsch, Zeev, Giladi, & Hausdorff, 2013). Because of this there are several factors that can diminish, sometimes severely, this ability, like muscular deformities, injuries or neurological diseases leading to an abnormal gait (Zheng, Yang, Wang, & Mcclean, 2009).

Strokes are one of the most common cause for this kind of problem, since many stroke survivors end up with gait impairments (Leuenberger, Gonzenbach, Wiedmer, Luft, & Gassert, 2014). Other causes can be multiple sclerosis, spinal cord injury or cerebral palsy (Zheng et al., 2009). For this kind of problem, the most recommended course of action is rehabilitation. This involves an assistive intervention that helps by reducing the lower-limbs loading, therefore, alleviating the joint pain and improving the functional ambulation, balance control, and gait's efficiency (Jung, Chae, Jang, & Park, 2012).

Some methods of rehabilitation that are currently used include walking aids such as canes, therapist-assisted manual training, treadmill with partial body weight support, functional electrical stimulation and virtual scenes that simulate walking in a slew of different environments. Walking aids are usually prescribed to patients with low level of movement impairment since they offer the least amount of support (Wilson, 2002). Most of these methods are however less than ideal since they require the patient to have therapy sessions in

specially crafted environment, which means that before improvements are seen, the patient is very limited on the amount of daily-life activities that can be done.

Due to the increasing integration of robotics in the field of medicine over the last few years, a larger number of people are given a chance to live a normal life without the limitations that impaired walking poses through rehabilitation with the help of several kinds of wearable assistive devices like prosthesis, orthosis and exoskeletons (Tucker et al., 2015).

The devices work by restoring a patient's ability to walk, in case of prosthetics, or, in the case of orthosis, helping a patient's mobility by providing an assistive force on the lower limbs joints. These have shown themselves to be very effective in rehabilitation treatments due to their customizability to the patient's degree of impairment, adaptability to the environment and ability to assess a patient's progress using an array of wearable sensors (Tucker et al., 2015).

But despite the considerable amount of research done with these devices there are still some issues that require solving to make the use of these kinds of devices as seamless and as unobtrusive to the user as possible. This involves motion recognition and prediction in an attempt to foster an assist-as-need rehabilitation/ user-centered rehabilitation.

1.2 Problem Statement

In an orthosis several systems must work in tandem to achieve a robust and safe functionality. One of these systems is the one that alters the way these devices operate to adapt to distinct locomotion modes, such as transitioning from a forward walk to ascending stairs which are distinct types of movement that require different control strategies (Li & Hsiao-Wecksler, 2013).

Most of the currently used approaches to this problem are based on human intervention, namely button presses or heel taps (Liu, Wang, & Helen Huang, 2016). These make sure the mode transition is performed only when the user wants but, because of this, they come with some drawbacks. Since they depend on the user, they make the transition less fluid as the system is not autonomous or reactive to its environment and thus can interfere in the normal locomotion. These kinds of systems also cannot be used by people who have difficulty pressing buttons or due to more severely impaired locomotion cannot tap their heels (Liu et al., 2016).

Another approach is the use of EMG signals from specific leg muscles in combination with mechanical measurements such as the amount of rotation of the hip to trigger these

transitions (F. Zhang & Huang, 2013). While mechanical sensors are inexpensive and can be fixed to the assistive device, EMG sensors are, however, hard to keep attached during the user's daily locomotion and require an expert's installation (Turker, 1993).

Another issue with this approach is that the tuning of the threshold values for both mechanical and EMG signal measurements requires medical expertise and a lengthy process since there is no automatic method to tune these values for all types of locomotion modes (Liu et al., 2016).

Other techniques that try to use only mechanical sensors for the prediction of locomotion mode transition have been tried. These however require the user to walk in a specific way, for example, always entering starting stairs with the healthy leg, and aren't usable for cases where the user requires assistance on both legs (Chen, Zheng, & Wang, 2014).

Because of these limitations, assistive devices can only be used in a clinical setting instead of the user's everyday life. It is then necessary to develop other solutions to the locomotion mode transition problem on assistive devices to ensure that these devices can be used with little medical supervision and with as little an impact on the user's gait as possible.

1.3 Goals and Research Questions

Expanding on the previously mentioned goal, this thesis intends to showcase the development of a machine learning pipeline that outputs a robust and adaptable classification model which, by using cheap inertial sensors worn by the user, has learned, through movement signals, how to predict the user's intention thus automating the orthosis locomotion mode change. It is also easily integrated in wearable devices and can reduce the amount of medical supervision needed during rehabilitation.

Therefore, the goals are to (i) analyze previous literature and identify the shortcomings of previous attempts and their respective solutions for the development of such models; (ii) design and conduct experiments involving healthy subjects walking in several terrain types (flat, ascending stairs, descending stairs, ascending slopes, descending slopes); (iii) specify and implement feature selection methods to analyze which are the most relevant features for the defined goal; (iv) explore and implement pre-processing methods to apply on the data for improved classification model building; (v) create a classification model building pipeline that can output person-specific models capable of recognizing the direction (forward, back, clockwise, anti-clockwise) of a human or robot subject as well as the type of terrain being

walked on and transitions from those terrains in a quick and efficient manner; (vi) compare four classification algorithms (DA, KNN, RF and SVM) commonly used in previous literature for the defined goal and evaluate the best one to be used in this case.

As such this document intends to answer the following questions:

RQ1: What techniques are most commonly used in previous literature?

RQ2: What is the most reliable metric to evaluate the obtained models?

RQ3: Which normalization technique is better for this kind of information and the selected model?

RQ4: Which is the feature selection procedure that strikes the best balance between computation time and model performance?

RQ5: Which classification algorithm is best for the recognition and prediction of locomotion modes?

RQ6: Which features have the most significant role for each classification objective?

RQ7: Is it possible to predict user locomotion intentions using only biomechanical data?

1.4 Thesis Outline

This document begins, in the chapter 2, by providing the reader with the necessary definitions for important concepts required for the full understanding of the procedures

Chapter 3 contains information about prior research done by other researchers on the topic.

In Chapter 4, a detailed description of the proposed solution and its associated procedure is explained.

Chapter 5 describes the tests performed using several algorithms of data collection, normalization and feature selection and extraction as well the testing of different classifier algorithms to find the best ones for the development of the solution for the studied problem. The chapter also includes the discussion of the obtained results for each of the phases.

Chapter 6 describes the procedure for the development of the model for recognition of locomotion mode and its results.

Chapter 7 describes the procedure for the development of the locomotion mode prediction model and its results.

Chapter 8 reports the conclusions drawn from this work and future challenges.

2. THEORETICAL BACKGROUND

In order to contextualize the problem and to provide the necessary concepts and definitions for the complete understanding of this document, a theoretical background section containing important notions. Several different notions from domains of data analysis and machine learning are important for the development of the solutions presented in this dissertation.

2.1 Assistive Devices

There are several kinds of assistive devices available for consumer use. These can take the shape of mobility, hearing or reading aids and help people with physical impairments in their daily activities. Some examples of mobility aids are wheelchairs, scooters, walkers, prostheses and orthoses. These are essential in both returning the ability to walk or rehabilitating a patient after the loss of motor functions. Since this study is focused on the development of a system to be possibly used in prosthesis and orthosis a more thorough explanation of both follows.

Orthosis are external exoskeletons that envelop the weakened limb and can be used in rehabilitation to restore a subject's motor strength. These can be passive or active whereas passive orthoses have no electrical components and thus are entirely dependent on the user, active orthosis incorporate electrical components such as motors to help the user with walking thus becoming less of a burden. An example of an active orthosis is depicted in Figure 1.

Figure 1 - Active orthosis

Prostheses are artificial mechanical limbs usually attached to a patient's stump that act as their missing limb. Just like orthosis these can be passive or active, distinguished by the same differences. However, contrary to orthosis which are mainly used for rehabilitation, prostheses are made to restore a patient's ability to walk or grab objects if it is a prosthetic arm. Figure 2 depicts an example of such a device.

Figure 2 - Prostheis

2.2 Machine Learning

Machine learning is a field of study that deals with algorithms that provide systems the ability to automatically learn and improve from experience without being explicitly programmed to (Samuel, 1959). This is done by analyzing data to find patterns or learn the its underlying model. There are three broad categories in this field (Russell & Norvig, 2010) which include supervised learning, unsupervised learning and reinforcement learning.

In supervised learning, data, composed of one or more predictor variables and followed by a response variable, is used to build a model that can output a correct prediction based on data it has never seen (Russell & Norvig, 2010).

The field of unsupervised learning uses techniques to infer unknown patterns or structure from the data, which is provided with only the predictor variables and no response variable (Russell & Norvig, 2010).

In the reinforcement learning field, through interaction with the surrounding environment, a system iteratively learns based on provided basic rules. This means that the system adapts to different circumstances by receiving input and performing actions that maximize its inherent reward it was provided thus finding a balance between exploration and its current knowledge (Russell & Norvig, 2010).

Supervised learning has two main purposes. One is classification where the goal is to obtain a model capable of correctly classifying unlabeled data. In this case the response variable (classes) takes discrete values that represent a category that the data point belongs to. An example of this would be a model that can classify pictures as that of a cat or dog. The other goal is regression which intends to model the underlying behavior of the seen data providing the expected response to an input after the system was trained using inputs with known responses. Here the response variable takes continuous values. For example, a regression model could try to predict a house's price based on its characteristics.

In unsupervised learning the most common uses are also two-fold. The first one is clustering, where the inputs are divided in classes that are previously unknown as opposed to classification where the data is labeled beforehand. This can be used as a precursor to a classification task. The second one is density estimation, where unsupervised learning techniques are used to estimate the data's unknown probability density function.

Under the context of supervised learning, predictor variables are usually referred to as feature variables since they represent a certain feature of the data. As such and for the remainder of this document this name will be used when referring to predictor variables.

2.3 Feature Engineering

Data availability has increased at a huge rate in recent years with the increased scrutiny our behaviors experience through the electronic devices used and the advent of cheap and reliable sensors that can be used for its collection. This can lead researchers in machine learning fields to compute as many features as possible from the data so that a robust model can be built. This is however the wrong procedure most of the time.

Data features can be characterized as three distinct types: relevant, irrelevant and redundant. Relevant features have an influence on the output and their role cannot be assumed by the rest. Irrelevant features are defined as those features not having any influence on the output, and whose values are generated at random for each example. Finally, a feature is characterized as redundant whenever that feature can take the role of another. This can be due to high correlation between two features. Some features can even be interdependent where they convey valuable information that might not be discernable if only one of them is included.

Due to the impact that irrelevant or redundant features can have, including a large number of features can make a model overly complex and computationally costly to build.

Because of this good feature engineering is the best tool to obtaining a small set of relevant features that can be used to create a more understandable and accurate model. However, this requires a very deep understanding of the problem which may be impractical to obtain. When this is done under the context of gait recognition it usually means obtaining peak values and magnitudes of signals. However, this can be very subjective and introduce many correlated and redundant features.

It is always wiser to select features through a careful study of the problem as part of the data pre-processing. This is true even with the possible use of techniques for dimensionality reduction since the use of these techniques can sometimes ignore an important feature or cause overfitting.

2.4 Data Pre-processing

Before using data in a machine learning algorithm, it is usually performed pre-processing to remove outliers and make it more usable. The phases for supervised learning are, as defined by Kotsiantis et Al. (Kotsiantis, Kanellopoulos, & Pintelas, 2006), Instance Selection and outlier detection, Missing Feature Values Removal, Discretization, Data Normalization and Dimensionality Reduction where both Feature Selection and Feature Extraction algorithms are included. The techniques used on each of these phases are described below.

2.5 Data Normalization

Data normalization's designation is not agreed upon and can have many different meanings (Upton & Cook, 2014) but in the context of this document it is assumed as the shifting or scaling of datasets bringing its variables to a common scale, so they can be more fairly compared. Before the building of classification models, it is customary to perform this on the used data since many of these models are sensitive to features with very disparate ranges. This, however, depends on the data and the model that is used (Kotsiantis et al., 2006).

There are several commonly used normalization techniques. Some of the most common are centering, scaling, standardizing and min-max scaling. Centering involves subtracting the mean of the data so that the mean is null, and the data is centered at 0. This formula is represented in Equation 1.

$$x' = x - \mu \quad (1)$$

where x is the original variable value, μ is the variable mean and x' is the normalized value.

Scaling involves dividing the data by its standard deviation thus turning the values into the amount of deviations from the mean. This can help ensure different datasets are on the same scale. This formula is represented in Equation 2.

$$x' = \frac{x}{\sigma} \quad (2)$$

where x is the original variable value, σ is the variable standard deviation and x' is the normalized value.

Standardizing is the most commonly used method and involves centering-then-scaling. In clustering (distance-based) analyses, standardization may be especially crucial to compare similarities between variable based on certain distance measures (these include KNN and SVM). This formula is represented in Equation 3.

$$x' = \frac{x - \mu}{\sigma} \quad (3)$$

where x is the original variable value, μ is the variable mean, σ is the variable standard deviation and x' is the normalized value.

Min-Max Scaling involves converting data into a range of values. This is usually used for algorithms that only work with data on a certain range of values like neural networks. This formula is represented in Equation 4.

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}(b - a) + a \quad (4)$$

where x is the original variable value, $\min(x)$ is the variable's minimum value, $\max(x)$ is the variable's max value, a is the new minimum, b is the new maximum and x' is the normalized value.

2.6 Dimensionality Reduction

If feature engineering is not possible or even after a properly done feature engineering one might still want to further reduce the number of features used. The field that deals with this problem is called dimensionality reduction.

The most common goals for the use of these techniques are reducing the number of features, to reduce overfitting and improve the generalization of models, and gaining a better

understanding of the features and their relationship to the response variables (Kotsiantis et al., 2006).

These two goals are often at odds with each other and thus require different approaches: depending on the data at hand a dimensionality reduction method that is good for one goal is not necessarily good for the other. This happens because certain methods may ignore correlation between features and pick either the correlated or the original feature ignoring the other. This makes data interpretation harder since one might be led to think that the feature that was not selected does not have a strong relationship to the response variable. If the goal is reducing the number of features this is not a problem and it is usually one of the goals to remove correlated features. The field is divided in two major categories: feature selection and feature extraction. These are explained in the following sections.

2.6.1 Feature Selection

Feature selection employs techniques that automatically select the most relevant features or at the very least rank them in accordance to their importance for use during model construction. There are three kinds of methods currently used for feature selection. These are Filter Methods, Wrapper Methods and Embedded Methods (Guyon & Elisseeff, 2003).

Filter methods are techniques that use statistical tests such as correlation tests, to rank features according to their importance. These are independent of the classification model and robust to overfitting while being computationally fast. However, these are prone to selecting redundant features and since they are not model-based they can lead to worse performance (Guyon & Elisseeff, 2003).

There are several examples of filter methods and some are described in the following text.

Pearson's Correlation (r) is used as a measure for quantifying linear dependence between two continuous variables. Its value varies from -1 to +1. This is used to compare two features and study if they might be related and thus revealing one of them as redundant (Hill & Lewicki, 2006). There are a few problems with this method. As the number of variables increases, the data storage requirement for saving these coefficients increases as (nearly) the square of the number of variables. For relationships that are non-linear, r is not a very good indicator of correlation. One must arbitrarily decide on the cut-off value of r after which a variable is considered relevant.

The Welch's t-test is an adaptation of the more common Student's t-test but is more generally applicable and is used to compare the same variable in two distinct groups to assess if their means are the same. It is a statistical hypothesis test where the test statistic follows a Student's t distribution if the null hypothesis is supported. This test is applied when the test statistic follows a normal distribution and the value of a scaling term in the test statistic is known. If the scaling term is unknown, it is then replaced by an estimate based on the available data. The test statistic follows a Student's t-distribution (Hill & Lewicki, 2006).

ANOVA stands for Analysis of variance. It is operated using one or more categorical independent features and one continuous dependent feature. It provides a statistical test of whether the means of several groups are equal or not. In the context of classification, this can be used by comparing the set of samples from each class within a feature vector and verifying if they are different enough. This test is better for the comparison of various groups than the t-test that, while less computationally intensive, is only useful for the comparison of two groups (Hill & Lewicki, 2006).

The Chi-Square test is a is a statistical test applied to the groups of categorical features to evaluate the likelihood of correlation or association between them using their frequency distribution (Hill & Lewicki, 2006).

The ReliefF algorithm estimates the quality of features according to how well their values distinguish between instances that are near to each other. Given a randomly selected instance *i* from class *L*, ReliefF searches for *K* of its nearest neighbors from the same class called nearest hits *H*, as well as *K* nearest neighbors from each of the different classes, called nearest misses *M*. It then updates the quality estimation W_f for feature *f* based on their values for *i*, *H*, *M*. If instance *i* and those in *H* have different values on feature *f*, then the quality estimation W_f is decreased. On the other hand, if instance *i* and those in *M* have different values on the feature *f*, then W_f is increased. The entire process is repeated *n* times which is set by users. ReliefF is a general and successful attribute estimator and can effectively provide quality estimates of attributes in problems with dependencies between attributes. However, ReliefF does not explicitly reduce the redundancy in selected genes (for example, the first three features in the ranking might be redundant being as useful as if only one of them was used) (Yi Zhang, Ding, & Li, 2008).

The mRMR (minimum-redundancy maximum-relevancy) algorithm is similar to the ReliefF method but besides selecting features that have the highest relevance with the target class labels it also selects those that are minimally redundant, i.e. selects features that are maximally dissimilar to each other. This creates a ranking that is, unlike the one obtained

from ReliefF, immune to redundant features although it is more computationally expensive (Yi Zhang et al., 2008).

The abundance of different filter methods, each with their own pros and cons, makes picking between them very dependent on the purpose and data use. The mRMR algorithm is however generally regarded as one of the best for ranking features (Yi Zhang et al., 2008).

Wrapper methods use the classification model to obtain the best set of features by creating several combinations of features, using them to build a model and evaluating its' performance to assign a score to the combination. These methods are tailored to the used algorithm and thus can be used to create a model with better performance. However, these can be very computationally expensive since they involve performing many model building procedures to test several feature combinations (Kohavi & John, 1997). Some examples of wrapper algorithms follow.

The Exhaustive Search algorithm builds models with every possible combination of features and output the feature vector that leads to the best performing model (Kohavi & John, 1997).

The Forward Selection algorithm is an iterative method in which we start with having no feature in the model. In each iteration, we keep adding the feature which best improves our model till an addition of a new variable does not improve the performance of the model (Kohavi & John, 1997).

In the Backward Elimination algorithm, we start with all the features and removes the least significant feature at each iteration which improves the performance of the model. We repeat this until no improvement is observed on removal of features (Kohavi & John, 1997).

The Recursive Feature elimination algorithm starts by creating a model with all features pruning the worst one based on its coefficient. It does this until all features are exhausted. It then ranks them according to their order of elimination. As such, it is a greedy optimization for finding the best performing subset of features that requires a model which assigns weights to features (Kohavi & John, 1997).

Genetic Algorithms are similar to the brute-force search technique in the sense that they test several sets of features obtained randomly by building models with each of those sets and testing their performance. It starts with a user-defined number of random sets (population) where each set is a vector of binary values (as many as the features) indicating if that feature is selected or not. Each one of these is called genome. Every set in the population is tested for its fitness value, which in this case is the accuracy of the model trained with those

features. The *n* best ones (number of elites, also defined by the user) are passed to the next iteration while the other genomes' probability of being used on the following generation is defined by their fitness value. The selected genomes are used to generate the new population through mutations (switching randomly the values of the genome to 1 or 0) and crossover (swapping values from two different genomes). This goes on for a user-defined number of iterations (generations) or until too many generations with the same best results pass (this parameter is also user-defined). Finally, the genome corresponding to the best fitness value is returned (Kohavi & John, 1997).

Embedded methods perform feature selection as part of the model building procedure making them much less computationally expensive then the wrapper methods. They try to combine the best parts of filter and wrapper methods. The most common type of embedded feature selection methods are regularization methods. They work by penalizing large coefficients in a model making it simpler, that is decreasing the influence that certain features have in the model (Guyon & Elisseeff, 2003).

2.6.2 Feature Extraction

Algorithms that try to find the optimal linear or non-linear combination of original features to reduce the dimensionality of the final problem are called Feature Extraction Algorithms. Feature Extraction is sometimes referred to as feature transformation or even feature construction.

A commonly used technique in this field that yields uncorrelated, linear combinations of the original N features is Principal Component Analysis (PCA). PCA transforms the features space into uncorrelated linear combinations of the original features that explain the most variability of the data. These new features are called principal components and are the ones that will later be used for the model building (SAS, 1989). After their calculation the top components are selected based on one of several different criteria (Ledesma & Valero-mora, 2007). Some of these criteria are described as follows.

Horn's parallel analysis criteria (PA) is a Monte-Carlo based simulation method that compares the observed eigenvalues with those obtained from uncorrelated normal variables. A factor or component is retained if the associated eigenvalue is bigger than the 95th percentile of the distribution of eigenvalues derived from the random data. PA is one of the most recommended rules for determining the number of components to retain. Parallel

analysis is partially sensitive to sample size in that for large samples the eigenvalues of random factors will be very small (Ledesma & Valero-mora, 2007)..

The Kaiser criterion is to drop all components with eigenvalues under 1.0 – this being the eigenvalue equal to the information accounted for by an average single item. The Kaiser criterion is not recommended when used as the sole cut-off criterion for estimating the number of factors as it tends to over-extract factors. A variation of this method has been created where a researcher calculates confidence intervals for each eigenvalue and retains only factors which have the entire confidence interval greater than 1.0 (Ledesma & Valero-mora, 2007).

The Scree plot criteria involves plotting the components as the X axis and the corresponding eigenvalues as the Y-axis. As one moves to the right, toward later components, the eigenvalues drop. When the drop decreases into a less steep decline and the trend starts resembling an elbow, Cattell's scree test says to drop all further components after the one starting the elbow. This rule is sometimes criticized for being amenable to researcher-controlled "fudging". That is, as picking the "elbow" can be subjective because the curve has multiple elbows or is a smooth curve, the researcher may be tempted to set the cut-off at the number of factors desired by their research agenda (Ledesma & Valero-mora, 2007)..

The variance explained criteria is very common and consists of keeping enough factors to account for 90% (sometimes 80%) of the variation. Where the researcher's goal emphasizes parsimony (explaining variance with as few factors as possible), the criterion could be as low as 50% (Ledesma & Valero-mora, 2007).

There is also an extension to this method called kernel PCA (kPCA) which allows for the construction of higher-dimension combination of features and thus tackle more complex problems (SAS, 1989).

It is worth considering that the PCA algorithm however may not be useful since the largest variance of the data does not necessarily represents the most discriminative information and the transformed feature vector cannot be directly translated to the features needed to get the chosen explained variance (SAS, 1989).

2.7 Classification Model Building

Machine learning can take the form of supervised or unsupervised learning, when dealing with data labeled with a class for the creation of predictive models or unlabeled data used for pattern search respectively (Alpaydin, 2010).

Over the years different algorithms were developed for the creation of models that can tackle these problems which can be applied according to the circumstances of the required model. The most used algorithms are Neural networks, K-nearest neighbors (KNN), Support Vector Machines (SVM), Discriminant Analysis (DA) and Random Forests (RF). Some of these algorithms are described in the following section.

K-nearest neighbors (KNN) is one of the simplest classification algorithms. Training this model is simply storing every train instance so that during classification it can compute the closest instances to the data point that is to be classified (the number of closest neighbors to which the point is compared to, designated *k* is defined by the user or by parameter tuning) and by using a majority vote on the neighbors' classes it obtains the new point class.

If weighted KNN is used, then the further the distance between an instance to be classified and its neighbor less influence that neighbor has on the voting (this is usually more robust than regular KNN). Several distance metrics can be used including Euclidean metric (L2 norm), which is he the most commonly used, City block metric (L1 norm), Chebyshev metric and others (Altman, 1992). An example of such a model is presented in Figure 3.

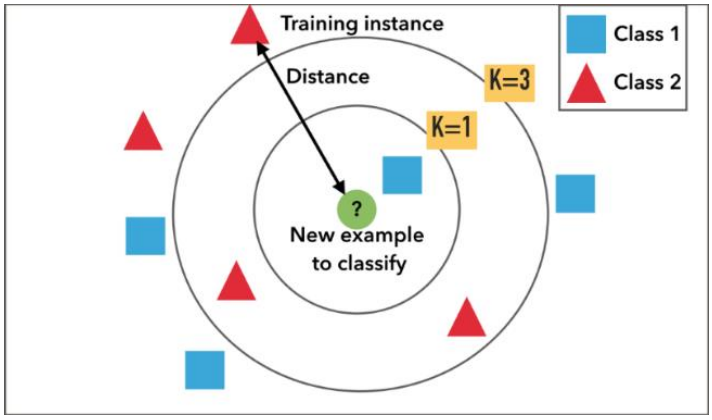

Figure 3 - KNN-based classification example

Support Vector Machines (SVM) algorithm tries to find a line or hyperplane with an associated margin that can separate the different data classes. It can also be used with different kernels. These project the data onto a new feature space to make the classes more separable and can be linear, polynomial and gaussian among others. Two hyperparameters of this algorithm is *C* and sigma, a parameter specific to the use of the gaussian kernel and therefore not used in other cases. *C* works as a regularization parameter which means that a larger *C* creates a model with lower bias but high variance (prone to overfitting) and with a

smaller *C* a model with higher bias but lower variance (prone to underfitting). The sigma parameter influences how smoothly the features vary which means that for a large sigma the feature will vary more smoothly meaning higher bias and lower variance. In contrast, a small sigma means a sharper variation of the features and thus a lower bias but higher variance.

SVM is usually a binary classifier but using a "one-vs-one", where a model is created for every combination of two features, or a "one-vs-all" strategy, where a model is created for every class where the opposite class is composed of all the other classes' data points, it is possible to overcome this limitation (Hsu, Chang, & Lin, 2008). An example of the hyperplane that an SVM algorithm might calculate is shown in Figure 4.

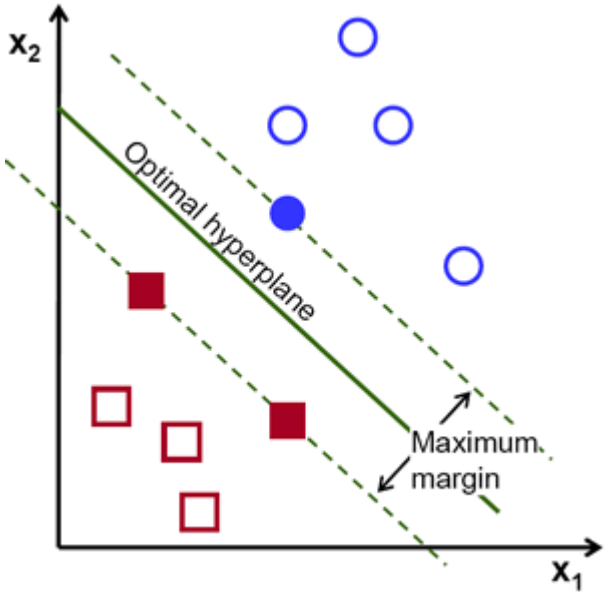

Figure 4 - SVM Hyperplane example (the axes represent the feature values and each shape is a different class)

Discriminant analysis algorithm is used to find a combination of features of a certain order (linear, quadratic, cubic…) that best separates the data based on two or more classes of a response variable. Like the PCA principle, it does this by attributing coefficients to each of the original features to obtain a hyperplane that divides the data according to its classes (Haghighat, Abdel-Mottaleb, & Alhalabi, 2016).

Random forests are based on the concept of decision trees, rule-based classifiers that use the feature values to create a tree of decisions where the last step is the classification result. Random forests are an ensemble model constituted from a number of these decision

trees built from random subsets of data and features creating an ensemble model that is very robust. The number of decision trees is specified by the user (Denisko & Hoffman, 2018).

The KNN and SVM are some of the most widely used algorithms. The KNN algorithm builds simple models that are easily interpretable and can be used in a lot of different situations while requiring few computational resources making it one of the most used for model building. On the other hand, SVM models are more complex and require more computational resources for both training and classification but it is much more versatile than the KNN algorithm and so it yields better results most of the time.

2.8 Model Validation

After a model is built its generalization performance, or how well it can lead with unseen data should be calculated, and this is referred to as model validation. There are many different techniques that can be used for this and depending on the desired goal many resulting metrics which range in both interpretability and what they represent.

2.8.1 Validation methods

Some of the most well-known validation methods are Holdout, Bootstrapping and *K*-fold cross-validation. These are explained in more detail further on.

In the Holdout validation test the data is split into two groups. One of these groups consists of a user-defined percentage of data which is used for validation while the rest is used for training the model, usually with 70% for training and 30% for validation (Kohavi, 1995). This can be repeated for more robust results, although, while in principle the averaged test errors on the repeated hold-out sets are superior to a single test error on any particular test set, it still has one drawback: some data rows are used in multiple test sets while others are never used for testing. Because of this, the errors made on those repeated rows have a higher impact on the test error. This creates a selection bias, which is undesirable.

The Bootstrapping validation method creates several datasets called bootstrap samples from the original dataset by randomly placing instances into the new datasets (these instances can be repeated). The number of bootstraps is defined by the user, but an often-recommended number of bootstrap samples is 1000, although this varies for every problem. A model is then trained with each of these samples and used to classify the original dataset. By averaging the

performance of these models, it is possible to obtain a good estimate of their generalization performance (Kohavi, 1995).

K-fold cross-validation (*k*-fold CV) is the most used and most recommended method for model validation. It operates by splitting the data into *k* folds, a user selected number, and training a model on all but one of the folds, which is used for testing. This is done until every fold was used for testing purposes. The performance is then obtained by averaging the results of each model. When *k* is equal to the number of observations, then you have leave-one-out cross-validation (LOO) since only one observation is used (Kohavi, 1995).

To reduce the inner variance that the random data splitting may cause across folds it is recommended to use repeated k-fold cross-validation (averaging the results of each one of the procedures to get a good stable measurement of the model's performance). Stratification is also recommended meaning splits are created with the same proportion of each class label instances as the original dataset. Otherwise, some test sets may not include observations from all classes.

The goal of the cross-validation procedure plays a role on how it should be done. According to Zhang et Al. (Yongli Zhang & Yang, 2015), for predictive performance estimation, LOO or repeated 50 and 20-fold CV are best but for Model selection, repeated 2-fold CV is best for identifying the best candidate model since it inflates the error rate of the models allowing for a better comparison. It is suggested that the minimum number of repetitions should be 10 to 20 times.

This validation method gives a pessimistically biased estimate of performance, that is, lower than the true value because most statistical models will improve if the training set is increased. It is important to note that cross-validation validates the entire model building process (including hyperparameter optimization and feature selection) so the entire process must be done inside each cross-validation fold for a better estimate of the model's generalization performance (Cawley & Talbot, 2010).

2.8.2 Performance metrics

Most of performance metrics are obtained from a confusion matrix, which describes the number of instances predicted as a certain class as well as the instance's true class. The confusion matrix has *n* rows and columns, where *n* is the number of class labels. The rows represent the true classes of the instance while the column the predicted class. Each cell contains the number of instances classified as the respective column class label on the row

corresponding to the instances' true class label. An example of a confusion matrix is shown in Figure 5.

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

Figure 5 - Example of a confusion matrix

Each part of the confusion matrix has a specific designation that has a key role in the calculation of the performance metrics. These designations are true positives (TP), false negatives (FN), false positives (FP) and true negatives (TN). Figure 6 shows the meaning of these designations.

		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Figure 6 - Meaning of each designation in relation to class label P

Some of the most used metrics are Accuracy, Precision, Sensitivity, Specificity, F-1 score, Matthews correlation coefficient (MCC), Area under Curve (AUC). All but the Area under Curve can be obtained from the confusion matrix. These metrics and their formulas are detailed in the following paragraphs. The numbers of the formulas are the confusion matrix designations (TP, FP, TN and FN).

Accuracy consists of the percentage of correctly classified feature vectors. It is the most widely used although sometimes erroneously since it should only be used when the number of classified instances is the same or close to the same for each class label. Its formula is expressed in Equation 5.

$$\frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

Precision is the ratio of true positives amongst all positive predictions (true and false). Its formula is expressed in Equation 6.

$$\frac{TP}{TP + FP} \tag{6}$$

Sensitivity is the true positive rate. This is the proportion of positive cases that are correctly identified as such. Its formula is expressed in Equation 7.

$$\frac{TP}{TP + FN} \tag{7}$$

Specificity is the true negative rate. This is the proportion of negative cases that are correctly identified as such. Its formula is expressed in Equation 8.

$$\frac{TN}{TN + FN} \tag{8}$$

F-1 score is the harmonic mean of the precision and sensitivity. It is widely used on the natural language processing field but there are better alternatives for machine learning since this measurement does not take into account true negatives. Its formula is expressed in Equation 9.

$$\frac{2 * TP}{2 * TP + FP + FN} \tag{9}$$

Matthews correlation coefficients (MCC) is a correlation-based metric that is widely used, and generally perceived as the best one, for cases where the number of instances of each class are very different (unbalanced) since under these cases other metrics yield improper values. This metric has some properties that facilitate its interpretation such as the metric being 0 value when half of the instances were correctly classified, but from its value it is hard

to discern exactly which class labels were wrongly classified the most. This makes MCC worse than accuracy for balanced cases. Its formula is expressed in Equation 10.

$$\frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \quad (10)$$

AUC, or Area under the curve, is the integral value of a ROC space. A ROC space is defined by the false positive rate and true positive rate as *x* and *y* axes, respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs). This metric has been shown to be a poor representative of model performance however (Lobo, Jiménez-valverde, & Real, 2008).

3. STATE OF THE ART

Several researchers have performed studies into locomotion mode prediction and recognition in the context of the human gait. Many of the more comprehensive studies were done with subjects using a powered prosthesis, but due to the similarities between the gait monitoring tools used by both systems these studies' conclusions can also be applied to orthosis (Tucker et al., 2015). This section describes previous literature regarding efforts to solve the tackled problem.

Jang et al. (Jang, Kim, Lee, Lim, & Shim, 2015) developed a FSM (fuzzy inference system) that could **recognize between three different locomotion modes (stair ascent, level ground walking and stair descent)**.

Three **healthy male subjects** (age 38.7 +- 5.7 years, height 173 +-3.6 cm) participated. They were instrumented with an **angle sensor** on **the sides of the hips** as well as **an IMU senso**r on **the lower back** for foot contact estimation. The sensors were sampled at 100 Hz.

Subjects were required to walk along walking environments at their preferred walking speed. Firstly, they walked along level environment freely, and they stopped in front of the staircase before ascending stairs. Then they ascended several stairs and stopped for a moment. They turned 180 degrees and stopped for a moment before descending stairs. Finally, they descended several stairs and stopped.

Retrieved features included the **left and right hip angle as well as the absolute difference between the two**, obtained when foot contact occurred.

At every foot contact the state-machine would use the feature values and through defined rules identify the current locomotion mode.

Validation was done by using the algorithm to identify the subject's trials. The used metric was accuracy (percentage of correctly classified steps). Results were **98% for stair ascent, 95% for stair descent, 99% for level ground walking**.

Li et al. (Li & Hsiao-Wecksler, 2013) developed a threshold FSM that could recognize between **five different locomotion modes (level walking, stair ascent, stair descent, ramp ascent, ramp descent)**.

The test subjects were **five healthy men** wearing a prosthetic foot with an average age of 23.4 years, average weight of 82 kg and average height of 178.6 cm. They were equipped

with an **IMU sensor on the medial side of the prosthetic foot** under the ankle that was sampled at 200 Hz.

The protocol was divided into a preliminary training session to collect data for threshold tuning and an experimental session to evaluate the algorithm. The preliminary training session consisted of a subject performing five trails in five different modes (level, stair ascent/descent, ramp ascent/descent) and the experimental tests consisted of the subjects walking on a mixture of terrains on four different walking environments.

The calculated features were the **vertical position change and foot pitch angle**. The vertical position was sampled at every heel or toe strike and compared with the previous one to obtain the change and the foot pitch angle was sampled at every foot flat. Both values were compared to predefined thresholds to identify the mode.

To evaluate the algorithm the percentage of correctly recognized steps was used. Results **were 98.3% for outdoor one stair traverse, 97.2% for outdoor two stair traverse, 99.5% for indoor two stair traverse and 99.1% for ramp**.

Chan et Al. (Chan et al., 2014) developed a classification model for recognizing **stair ascent or stair descent** by comparing Multilayer perceptron layer, K-Star, Support Vector Machines, Naïve Bayesian, Decision Trees and random forests.

Test subjects were **thirteen healthy young adults** with average age of 27.7, average weight of 68.8 kg and average height of 1.71 m and 12 older adults with an average age of 70, average weight of 67.5 and average height of 1.66 m. They wore a **triaxial accelerometer and gyroscope system attached to the lumbo-sacral joint** and the data was sampled at a rate of 100 Hz.

The experimental procedure consisted in the subjects walking up and down a 13-step staircase. These trials were repeated four times by each subject.

The retrieved features were the **cadence, root mean square and the integral of power spectral density of the vertical, medio-lateral and anterior-posterior directions, symmetry in the vertical and anterior-posterior directions, stride regularity in the vertical, medio-lateral and anterior-posterior directions and step regularity in the vertical and anterior-posterior directions**. These were obtained from each full trial and labeled with the corresponding locomotion mode and age group.

The classifiers were used to discriminate between going up or down stairs and between young or old subjects. A correlation-based feature selection method and a forward sequential selection method were also compared.

Classifiers were evaluated using 10-fold cross-validation and based on four measures: accuracy, specificity, sensitivity, and area under the curve. The best results were obtained when using the correlation-based feature selection algorithm. In discriminating younger from older adults, the measurements were 84.9% accuracy, 0.89 AUC, 82.2% sensitivity and 87.5% specificity for young adults and 87.5% sensitivity and 82.2% specificity for older adults. When discriminating climbing upstairs from climbing down stairs the results were: **96.8% accuracy, 0.995 AUC, 95.7% sensitivity and 97.8% specificity for climbing upstairs and 97.8% sensitivity and 95.7% for climbing down stairs**. The best model was MLP.

Leuenberger et Al. (Leuenberger et al., 2014) developed a method to identify locomotion modes, namely**, level walking, stair ascent and stair descent**. This was done by creating a radial basis kernel SVM classifier to identify walking phases (walking or other) followed by a KNN classifier used to discriminate the locomotion mode.

Data was obtained from two different sets of stroke patients (24 in total). The first set consisted of **12 subjects (3 female, 9 males, age 63.5 +- 15.1 years)** and the second consisted of another **12 subjects (6 female, 6 male, age 57.3 +- 17.3)**.

10 degrees of freedom **IMU sensors placed on the wrists, shanks (laterally) and on the trunk** provided the necessary data at a sampling rate of 50 Hz. These sensors included 3 DOF accelerometers, 3 DOF gyroscopes, 3 DOF magnetometer and a barometer.

Subjects from the first set were asked to perform several different indoor activities standing or sitting down while walking between each activity. Subjects from the second set were asked to walk three rounds of 300m on a park that included level walking, stair ascent and stair descent.

The calculated features used on the SVM classifier were the **signal energy, range, variance, as well as 10th and 90th percentiles of the acceleration magnitude as well as the skewness, kurtosis, maximum value and scale of the wavelet.** Feature selection using ReliefF was used to select the 18 most important ones.

The features used on the KNN classifier were the **slope of the altitude signal, the mean of the acceleration magnitude and variance, the range, as well as the 10th and 90th percentile of the magnitude of the acceleration.** No feature selection was used.

All features were retrieved from 7.5 second windows with 50% overlap and labeled with the activity with longest duration that happened during that window. If more than one activity took place, then it was also marked as a transition.

The SVM model was trained with a radial base function kernel and the first dataset where transition and activity windows were labeled as ‘other’ while all walking activities were labeled as ‘walking’.

The KNN model was trained with the second dataset, being only used when the SVM model classified a window as walking, and transition windows were not used in the training of this classifier.

Repeated holdout (20 repetitions) with 80% of data for testing and 20% of data for training was used. Classifiers were evaluated using sensitivity and specificity. The best specificity and sensitivity for the SVM classifier was obtained for the combination of both shanks and trunk sensors (95.9% sensitivity and 96.3% specificity), and the best features were kurtosis and skewness of the power distribution as well as the range of the acceleration magnitude. In the case of the KNN classifier the best specificity and sensitivity was obtained with the use of the shank sensors (**93.4% sensitivity and 85.7% specificity for level walking and 96.1% specificity and 87.1% sensitivity for stair ascent and descent**). The highest ranked feature was the altitude signal from the barometer.

Tkach et. Al (Tkach & Hargrove, 2013) used an LDA machine learning algorithm to investigate which kind of sensors yielded the highest accuracy (EMG or mechanical) when predicting locomotion mode transitions (**level-ground walking to stair ascent, stair descent, ramp ascent and ramp descent and vice-versa**).

5 unilateral trans-tibial amputee subjects (4 male, average age: 36 +- 7 years), of which 2 had an amputated right limb and 3 had an amputated left limb, participated in the study.

The subjects were fitted with an ankle-foot prosthesis and data was retrieved from **EMG signals of four below-knee leg muscles on the amputated side** as well as mechanical sensors on the prosthesis that included a **3-axis IMU** (accelerometer) in the **thigh and an encoder on the ankle.**

The subjects were instructed to walk at a comfortable speed on three types of terrain, level ground, stairs and ramps, performing the necessary transitions between them.

From the EMG sensors signals the retrieved features were: Mean absolute value, zero crossings, slope sign changes, and waveform length time-domain. From the mechanical sensors the extracted features were: prosthesis vertical acceleration, pitch angle, pitch velocity, velocity angle, accelerometer vertical and linear axes and angular acceleration in the sagittal and vertical planes (Mean and standard deviation).

All features were calculated from three types of window: 250ms window occurring 50ms before the toe-off (TO) of the mode transition step (preTO), 250ms window occurring 50ms after the TO of mode transition step (postTO), and 250ms window comprised of two 125ms windows, each positioned 50ms prior and post the TO of the mode transition step (acrossTO). Only steps from the prosthetic leg were used. These were labeled with the respective locomotion mode transition.

The LDA model was evaluated with leave-one-out cross-validation. The lowest error when classifying transitions **was 2.3% when using both EMG and mechanical sensors with the acrossTO window**. When using only mechanical sensors the results were: 12%, 8.5% and 7% for preTO, postTO and acrossTO. When not including ramps every error rate decreased and the best window was the postTO window except for the fusion of EMG and mechanical sensors where acrossTO window was better. The use of only EMG sensors yielded the worst results.

Huang et Al. (Huang et al., 2011) developed a machine learning algorithm that could recognize six locomotion modes **(level walking, stepping over obstacle, stair ascent, stair descent, ramp ascent, ramp descent**) and predict five locomotion task transitions (**level-ground walking to stair ascent, ramp ascent, and stepping over an obstacle and stair descent and ramp descent to level-ground walking**), using LDA and RBF SVM algorithms.

Subjects that participated in the study were **5 transfemoral amputees** with ages: 51, 56, 26, 42, 57; weight: 80.3, 75.2, 53.4, 66.1, 75.8 and height: 177.8, 173.4, 160.2, 165.8, 175.3.

Data was retrieved **from EMG sensors and a 6-DOF load cell in the prosthesis** worn by human subjects at 1000 Hz. Gait phases were detected by insole sensors.

Subjects were instructed to walk at a comfortable speed in a course that began with level walking, transited to stair/ramp ascent, turned 180º, performed stair/ramp descent and switched back to level walking. On stepping over obstacle all subjects used the prosthetic leg as the leading limb. These tests were repeated 15 times.

Retrieved features were **the mean absolute value, number of slope sign changes, waveform length, and number of zero crossings from every EMG signal and the maximum, minimum and mean value of each direction of force from the load cell**.

The data was segmented in sliding windows of 150ms with a 12ms increment. These windows were labeled with the locomotion mode and gait phase that most of the window was on. During transitions the windows were labeled with the next locomotion mode after the start

of the single limb stance of the prosthetic leg right before it switches form one terrain to another.

The classifier had two levels: a classifier of gait phases followed by a phase specific locomotion mode classifier. A window majority vote was used to switch locomotion mode (15 or 5 votes depending on a rare transition, such as stair ascent to stepping over obstacle, being detected or not, respectively).

Classifiers were evaluated using leave-one-out validation. For evaluation purposes, data was divided into static states and transition states. Transition states were defined as starting at the initial prosthetic foot contact before switching the negotiated terrain and terminating at the end of the single-stance phase after the transition. In static modes accuracy was used to validate while in transition periods the number of missed transitions and the time to classify them before the critical time was used.

The critical time was defined as the time until which the transition had to be detected (beginning of swing phase when transitioning from level walking and heel contact when transitioning to level walking).

Best results were obtained with the SVM classifier with both EMG and mechanical data with accuracies from **91.8 to 100% in static states, 0 misclassified transitions and an average of 400ms before critical time of prediction time.**

Zhang et. Al (F. Zhang & Huang, 2013) studied which sources of information were most adequate for user intent recognition in a prosthesis when using a non-linear SVM algorithm to build a phase-dependent classifier followed by multiple locomotion mode classifiers (one for each phase). Studied intentions include **sitting, standing, level-ground walking, stair ascent, stair descent, ramp ascent and ramp descent**. This is a continuation of the previously analyzed article (Huang et al., 2011).

Experimental protocol was done with **two male and two female patients with unilateral transfemoral amputations** with an average age of 50.5 years, average weight of 69.2 Kg and average height of 168 cm.

Considered data sources were **EMG signals from some of the leg muscles, a 6-DOF load cell on the pylon of the prosthesis and two 6-DOF IMU sensors (accelerometer and gyroscope) affixed to the lateral side of the prosthetic socket and pylon** sampled at 100 Hz.

Subjects were asked to perform trials for each of the locomotion modes. For the task of level-ground walking, we instructed subjects to start from standing, walk along a straight

walk- way, and stop and stand. For sitting and standing tasks, subjects were asked to transition between sitting and standing. During collection of training data for stair ascent/descent or ramp ascent/descent, subjects switched from level-ground walking to stair ascent/decent or ramp ascent/descent and then switched back to level-ground walking.

From the EMG signals the retrieved features were: **mean absolute value, number of slope sign changes, waveform length and number of zero crossings**. From the load cell the **mechanical ground reaction forces and moments** were obtained. From the IMU sensors the retrieved features were **the linear accelerations, angular velocity and acceleration of the thigh segment on the three axis, knee angle, knee angular velocity, and knee angular acceleration**. For both the IMU sensor and load cell the extracted features were the mean, maximum and minimum values.

These features were obtained from 150 ms overlapping windows with an increment of 50 ms. These windows were then labeled with the corresponding gait phase and locomotion mode. During transitions the windows were labeled with the next locomotion mode after the start of the single limb stance of the prosthetic leg right before it switches form one terrain to another.

During classification a 5-point majority vote scheme was used. The methods of feature selection tested were mRMR, sequential forward selection and sequential backward selection, these were used to select the data sources (which provide several features). mRMR worked by ranking data sources according to their relevance (highest rank of the associate features) and these sources were added starting from the best one until accuracy was higher than 95% and there was not a higher number of missed mode transitions than the baseline (all data sources). Both forward and backward selections iteratively added or removed sources until no improvement was achieved.

To evaluate the model a new set of trials were done where subjects were asked to transition among seven task modes in a fixed sequence: sitting, standing, level-ground walking, stair ascent, level-ground walking, ramp descent, level-ground walking, ramp ascent, level-ground walking, stair descent, level-ground walking, standing and sitting. These trials were repeated 15 times. These trials were later repeated with only the selected subset of features to determine the efficacy of the selected best feature selection algorithm.

Two kinds of performance metrics were used, accuracy for static states and number of missed transitions and transition prediction time for transition states. Without feature selection the model had a **performance of 98% and 3 missed transitions** for each of the subjects, which included transitions from level-ground walking to ramp ascent and ramp

descent to level-ground walking. With feature selection, accuracy decreased to 95% due to this being the chosen acceptable threshold for drop in accuracy when using feature selection but the computational time halved.

mRMR was selected as the best method since it was more computationally efficient and yielded similar performances to the other algorithms. More than half of the selected features were EMG signals, with vertical ground force being constantly picked and thigh segment acceleration the only kinematic feature selected.

Novak et al. (Novak et al., 2013) developed an algorithm for the detection of **gait initiation and termination** during level walking using classification trees.

Ten male subjects with no physical or cognitive abnormalities that would affect gait participated in the study. Their age was 33.1 ± 13.2 years, their weight was 75.0 ± 7.4 kg, and their height was 174.1 ± 3.8 cm.

Data was retrieved at 100 Hz from two types of sensors**: inertial measurement units (IMU) placed on each foot, shank, thigh and upper arm and one in the back as well as pressure-sensitive insoles on each foot**.

The subjects were asked to perform 40 gait trials, 20 of normal gait and 20 of fast gait. In each of these trials the subject would stand still for a certain amount of time and then walked across the room, stopped for a while and returned.

The calculated features were **six joint angles and velocities (ankles, knees and hips) from the IMUs and the center of plantar pressures along the antero-posterior direction and the total vertical ground reaction force from both insoles**.

The classification tree would have to detect toe-off and gait onset for gait initiation, and if a step was the last step for gait termination. For gait initiation, each instance of these features was classified by a classification tree as before or after gait onset and if four samples were consequently classified as such gait onset is detected and a second classification tree would classify as after or before toe-off where four samples would also have to be classified as 'after toe-off' for it to be detected. For gait termination the features were obtained from a buffer (mean, median, standard deviation, range, skewness and kurtosis) that started at the beginning of the step (starting on heel strike). Several buffer lengths were tested.

The trees were evaluated using leave-one-out cross-validation. Two types of leave-one-out were used: within-subject and subject-independent where the subject-independent would exclude data from one subject instead of one trial. For gait initiation the percentage of trials with undetected events and the mean and median of the absolute time difference (AE)

between events and detection was reported for accuracy. For gait termination the percentage of correctly classified trials was used. **For gait initiation the percentage of trials with undetected events was under 3%**. The use of only IMU sensors or only insole sensors yielded similar results: **a median AE of 0.08s for gait onset and a median AE of 0.05s for toe-off.** Combining both types only decreased the median AE to 0.07s. Subject-independent results were worse (only the percentage of trials with undetected events remained the same). These results were similar for both normal and fast gait. For gait termination, using IMU sensors only yielded the best results with over 80% of trials correctly classified within-subject with a buffer shorter than the average step (84.2% with a 0.5 s buffer for normal trials and 81.9% with a 0.5s buffer for fast trials). After 0.75s buffer performance reached 100%. Subject independent results were worse.

Chen et Al. (Chen et al., 2014) developed an **LDA classifier** for the recognition of locomotion modes that included **level-ground walking, stair ascent, stair descent, stair descent, ramp ascent, ramp descent and standing as well as recognition of locomotion mode transitions**.

Seven able-bodied subjects participated in the research. They had an average age of 24.1 (±0.5) years, an average height of 1.71 (±0.02) m and an average weight of 72.0 (±2.5) kg.

The subjects were instrumented with **IMU sensors**, consisting of an accelerometer, gyroscope and magnetometer, **on one of the legs and foot pressure insoles in both feet**. The IMU sensors were placed on the **front of the thigh and back of the shank and foot.** All sensors were sampled at 100 Hz.

The experiment consisted of 40 trials where the subjects would walk through a course where they would start standing still, followed by level-ground walking, stair ascent, level-ground walking, ramp descent, level-ground walking, standing, turning back, standing, level-ground walking, ramp ascent, level-ground walking, stair descent, level-ground walking, standing, turning back and finally standing. These trials were divided into pairs where on one the subject would start with the instrumented leg (Set-A) and the other trial with the non-instrumented leg (Set-B).

The calculated features were the **maximum, minimum, mean value, waveform length, standard deviation and root mean square of the pitch angle, roll angle, accelerations in the two axes of the gait direction and pressure from both feet**. These were retrieved from sliding windows of sizes ranging from 100ms to 200ms with an

increment of 10ms. The windows were labeled with the gait phase (initial double-limb stance, single-limb stance, terminal double-limb stance and swing phase) and locomotion mode that more than half of the window encompassed.

A boundary was defined after which the locomotion mode changed during transitions. The boundary for the transition between stand to other locomotion modes was the moment when either foot left the ground. From other locomotion modes to stand, the boundary was the moment when the swing leg contacted the ground before standing still. For the rest of the locomotion transitions, the boundary was defined as the middle of the swing period of the leg which first left the previous terrains.

The classifier has two levels: a classifier of gait phases followed by a phase specific locomotion mode classifier. A majority vote approach with the last 5 windows was used for the classification where each window would have as much weight as the posterior probability of the classifier.

Performance evaluation was done using leave-one-out cross-validation and in order to evaluate it the gait period was segmented into transition periods and steady-state periods. Transition periods were defined as: in transitions from stand to other locomotion modes, the locomotion transition period began when either foot left the ground and ended at the beginning of the next SS phase. For transitions from other locomotion modes to stand, the transition period began when the swing leg left the ground and ended when the swing leg contacted the ground before standing still. The transition period lasted for the length of a swing phase. For transitions between other locomotion modes, the transition period began when the swing leg left the ground just before transition and ended at the beginning of SS phase immediately after transition. The other time periods were defined as steady-state periods. For steady locomotion periods average accuracy was used for evaluation. For transition periods the number of missed detections and the prediction time, or the time difference between the first correct decision and the transition's critical moment.

Critical moments were as the time until when the model must classify the new locomotion mode after a transition boundary. For transitions from other locomotion modes to stand, it was the time when the swing leg (either the measured leg or the unmeasured leg) contacted the ground before standing still. For the other locomotion transitions, the critical moment was the beginning of initial double stance (i.e., foot-contact) for the measured foot after the transition.

The results were best when the classifier was trained with both Set-A and Set-B with a 150ms window. Increasing the window size further did not yield significantly better results.

Accuracy results for steady periods were, on average**, 99.6%** and for transition **periods there were 0 missed detections and prediction times averaged 400ms** before the critical time except for ramp descent to walking and walking to ramp descent where prediction time was close or higher than the critical time. It was also verified that only 6 trial pairs (one from each set) were required for good classifier performance.

Novak et Al. (Novak, Goršič, Podobnik, & Munih, 2014) developed an algorithm for the **detection of turns** during the human gait where **threshold values** were used to detect turn onset and an **LDA classifier** for **the turn direction and amplitude.**

Ten unimpaired subjects (9 males, 1 female, mean age 30.9 years, standard deviation 4.3 years) and a **male transfemoral amputee** (66 years old, 180 cm) participated in the study.

They were instrumented with **nine IMU sensors: one on each foot, shank and thigh, one on the lower back, one on the upper back and one on the head**. These sensors had a gyroscope, accelerometer and magnetometer and were sampled at 100 Hz.

Each subject performed 49 trials at a natural walking speed. Each trial consisted on the subject standing still for 5 seconds, and then proceeding to walk forward until a turning zone after which they would either keep walking straight, turn left by 22, 45 or 90 degrees, or turn right by 22, 45 or 90 degrees. They would continue until the end of the marked travel path after which they would turn around and go back to the starting position. The 49 trials were divided into sets of 7 where the subject would perform each possible travel path.

The turn moment and direction were recorded using visual markers worn by the subject and two cameras for later data labeling.

The obtained features were angular velocity, angle and magnetometer output taken from the axis around which the turn was done. These were subtracted from the values of the start of trial and obtained from a single instance in time.

The algorithm would start by first detecting turn onset using the sensor signals and thresholds (a cost function was designed, and genetic algorithms were used to tune the algorithm's thresholds, which included orientation and/or angular velocity thresholds) followed by a classification model that would detect turn direction and amplitude.

The turn detection algorithm was validated using within-subject and subject-independent leave-one-out cross-validation. Metrics evaluated were the percentage of correct detections (CD), percentage of premature and late detections (PLD), percentage of false negatives (FN), percentage of false positives (FP) and mean difference between detected and

reference offset (DIFF). The classification model was evaluated using the percentage of correctly classified directions and amplitudes at the time of turn onset + 0, 0.1, 0.2, 0.3, 0.4 and 0.5 s (only for trials where a turn was detected).

The best results for the turn onset detection were obtained when using both orientation and angular velocity thresholds and an IMU sensor on the upper back for within-subject validation **(CD = 87.4; PLD = 10.4; FN = 2.2%; FP = 4.8; DIFF = 69)** and lower back for subject-independent validation **(CD = 77.3; PLD = 21.9; FN = 0.8%; FP = 8.1; DIFF = 56)**. For classification of turn direction the linear discriminant analysis algorithm performed better with the head IMU (100%) during all times (T; T + 0.5) followed closely by the upper back IMU (99%) also during all times when using within-subject validation while using subject-independent validation produced better results when using the head IMU (100%) followed by the lower back IMU (97% to 99% from [T: T + 0.5]). Turn amplitude detection using within-subject validation produced better results when using the head IMU (67% to 80%) but the upper back IMU produced better results at T + 0.5 (83%). With subject-independent validation the lower back IMU yielded the best results after T + 0.2 (62% to 80%) with the upper back IMU yielding the better results before T + 0.2 (54% to 62%).

Huihua et Al. (Huihua, Reher, Horn, Paredes, & Ames, 2015) developed a **neural network classifier** for **locomotion mode transition recognition** (**level-walking, stairs ascent and stairs descent**).

One male transfemoral amputee participated in the data gathering trials.

The subject wore a prosthesis with **embedded encoders** while the healthy leg was instrumented with an **IMU sensor** (gyroscope and accelerometer) on the **shin and thigh**. These were sampled at 200 Hz.

The subject was instructed to perform ten trials of each task (transitions from stair descent to level walking, from level walking to stair descent, from stair descent to stair ascent and from stair ascent to stair descent).

The obtained features were the **knee and ankle angle and angular velocity from both legs**, per instance. Since the prosthesis foot remained flat with the ground the ankle angle was calculated using that assumption.

These features were labeled as the starting locomotion mode until the swing knee angle passed a certain threshold after which it was labeled as the final locomotion mode. This threshold was based on the knee angle from the healthy leg.

The simplest neural network model was chosen to avoid overfitting. A locomotion mode would only be considered detected if it was continuously detected until a score reached a certain value.

The model was validated in an online fashion where the subject was asked to perform 14 trials that accounted for 56 transitions. **Only 1 transition was misclassified (from stair descent to level walking).**

Ngo et Al. (Ngo, Makihara, Nagahara, Mukaigawa, & Yagi, 2015) developed solutions to three different problems under the context of gait action recognition: **a signal segmentation procedure that is speed and intensity invariant, a way to deal with the inconsistency of the sensors orientation when it comes to IMU sensors and an algorithm to classify different locomotion modes.**

The signal segmentation procedure developed relied on the signal from a waist accelerometer to correctly segment steps at the time of the heel strike. This procedure is based on the signal energy and density of local feature points of the smoothed signals from all every accelerometer axis. The higher the energy and density of the feature points (e.g. peaks and valleys) the more likely an event of heel strike happened.

Inconsistencies of sensor orientation are dealt with by first applying a rotation matrix (obtained from the gravity vector) to the original coordinate system to align it with the gravity acceleration. To perform the yaw correction the signal is rotated until it resembles the signal from a signal gallery where signals are known to be properly oriented.

The classification algorithm used was a linear SVM to classify level walking, ascending stairs, descending stairs, ascending slopes and descending slopes.

460 people participated on the study, aged between 8 and 78, with a similar gender ratio.

They were instrumented with a waist belt containing an **IMU sensor on the left and right sides and on the back**. These were sampled at 100 Hz.

The subjects were asked to walk straight on flat ground, up stairs, down a slope, up a slope, down stairs, and walk straight out and these trials were recorded for later labeling.

The obtained features were the **dissimilarities between the acceleration and gyroscope values of a test signal when compared with a template signal for each class or locomotion mode**. These would be extracted by comparing a window composed of two consecutive steps with a similar template window of each class and averaging the 10 closest point distances between the two signals. This creates a feature vector composed of 2*n

features, where n is the number of classes thus creating a dissimilarity value between acceleration and angular velocity for all classes. These values were normalized by z-score and labeled according to the trial recorded video. No locomotion mode transitions were considered (transition steps were included in the signal corresponding to the mode being transitioned to).

The classification algorithm was evaluated using the trials of 231 people for training and the trials of 229 people for testing. These were picked randomly, and the used metric was the average recognition accuracy. **The average accuracy remained similar (0.93)** using the proposed method across all sensor-orientation differences whereas other methods decreased in performance the bigger the difference was**. By locomotion mode the average accuracy was 0.85 for level-walking, 0.99 for ascending stairs, 0.98 for descending stairs, 0.91 for ascending slope and 0.94 for descending slope**.

Liu et Al. (Liu et al., 2016) developed a system for the **recognition of locomotion modes** using information from the environment and an **LDA classifier.**

Six able-bodied subjects and a transfemoral amputee participated in the study (average age of 27.4 years, average weight of 173 +- 6 cm and average body weight of 78 +- 5 Kg).

These were instrumented with **foot switches** and an **IMU sensor** attached to a **laser distance meter** placed on the **waist** on the same side as the prosthetic leg if there was one. The laser distance meter was rotated 45 degrees below the x axis of the accelerometer which pointed forward and was perpendicular to the trunk. This constituted the terrain recognition module (TR). All subjects were also instrumented with EMG sensors on some of the thigh muscles and able-bodied subjects also wore the prosthesis using an adaptor. This prosthesis was equipped with an encoder on the knee joint for angle and angular velocity measurement as well as a 6-DOF load cell on the prosthetic pylon. This constituted the locomotion mode recognition module (LM). The terrain recognition module was sampled at 100 Hz and the locomotion mode recognition module was sampled at 1000 Hz.

Three experiments were conducted. On the first one only the TR module was used, and only able-bodied subjects participated. They were instructed to start standing and then walk on a predefined path consisting of a level-ground walkway, five stairs and a 10-foot ramp with a slope of 1:12. When the end was reached they had to turn around and come back to the starting position through the same path. These were repeated 30 times per subject where they were asked to walk at different speeds. Both the TR module and LM module were used

for the second and third experience. In the second experiment two able-bodied subjects and one amputee were asked to perform two sessions of trials: a calibration session, where data from all locomotion modes was collected in order to estimate classifier parameters and calibrate the TR module, and a testing session, where the subjects were asked to walk from a starting position, ascend a ramp or stairs, make a turn on the platform, and walk back to the starting position. Eight ramp trials and eight stair trials were conducted (16 trials total for each subject) during this session. All subjects could start the new mode with whichever leg they chose for these trials. Only the amputee was involved in the third experiment where he was asked to walk in an environment with office chairs, tables and other obstacles and where the subject was asked to walk towards a ramp or stairs and then walk around it before stepping on it. Eight trials were done under this experiment.

Obtained features were: from the LM module the **mean, absolute value, number of slope sign changes, waveform length and number of zero crossings of the EMG signals, the maximum, minimum and mean of each DOF of the load cell.** From the TR module the **maximum terrain height or depression as well as the signal-magnitude area from the IMU sensor.**

These were taken from a 50ms window with 50ms increment. Only the features from the LM module would be used with the classifier. The terrain height served to provide the classifier with the prior probabilities of each class (these were obtained through height thresholds and decision trees that first separated up terrain from down terrain and level-ground, which constituted the coarse output and then further divided into upstairs, up ramp and other or downstairs, down ramp and other, where other would be obstacles in the terrain, which constituted the refined output). The signal-magnitude are of the IMU sensor from the TR module served to detect turns also through thresholds, disabling the TR module and thus setting all probabilities to uniform when turns or standing was detected.

Features from the LM module were labeled according to the mode that most of the window was in (transitions were defined as such (critical time): from level ground walking to other LMs, it was defined as the last toe-off event from LG; from other LMs to level ground walking, it was defined as the first initial contact on the LG. All transitions were based on the prosthetic side for an amputee).

The performance of the TR module was assessed through the response time, that is, the time elapsed between the time when the terrain change was recognized and the critical time. The LM module was evaluated using classification accuracy for static periods and number of missed transitions for transitional periods. A transition was considered correct if it

was recognized and lasted 200ms. **The TR module had a performance of 99ms on average**. **The LM module had an average accuracy of 95% in static periods** with the refined output from the TR module being the best, although the difference to the coarse output were not statistically significant. During transition periods **no transitions were missed** by using TR information whereas six transitions were missed without it.

Young et Al. (Young & Hargrove, 2016) developed an **intention recognition system** that could deal with **level-ground walking, slopes, and stairs** using **LDA classifiers and Dynamic Bayesian Networks**. **Eight unilateral transfemoral or knee disarticulation amputees** (seven male and one female) participated in this study. Subjects varied in age (24–64 years), weight (62–112 kg) and height (1.65–1.87 m).

They were instrumented with **potentiometers and encoders on the prosthetic knee and ankle, an axial load cell also on the prosthesis** and a **6-DOF IMU on the shank**. These were sampled at 500 Hz.

Subjects performed 14 trials of steady-state level-ground walking. During these trials, the subject started from a stop, walked to the other side of the room, stopped and turned around, and walked back. Next, subjects performed 20 repetitions of a circuit involving level-ground walking and stairs. The subjects started from a stop, walked to the stairs, proceeded up the stairs in a step-over-step gait, walked forward on an elevated platform at the top of the stairs, and stopped and turned around. The subjects then returned by walking forward, going down the stairs in step-over-step gait, and walking forward on level ground. Subjects then performed a final set of trials in which they completed 20 repetitions of a similar circuit, but the stairs were replaced with a 10 ramp that was approximately 4.3 m long.

The obtained features were the **mean, standard deviation, minimum, maximum, start and end values from the knee and ankle angles, velocity and torque signals, the axial force signal and the shank's three-directional accelerations and rotational velocities signals**. These were extracted from analysis window of 300ms that ended at 0%, 25%, 50% and 75% of stance phase as well as 0%, 25%, 50% and 75% of swing phase where 0% of swing phase corresponded to toe-off and 0% of stance phase corresponded to heel contact. Each window was labeled with the corresponding locomotion mode assuming the prosthesis transitioned at heel contact when the subject began walking on or off the ramp and descending stairs. For stair ascent, the transitions were defined at toe-off of the prosthesis before the subject started on the first stair.

Four types of model architecture were tested: the baseline configuration, two LDA classifiers, one at heel contact and toe-off, trained with the data where transitions were labeled as the mode being transitioned to; time-history configuration, eight LDA classifiers (one at each gait event) that were trained in the same fashion but where prior class probabilities were calculated through the application of the Bayes' law and prior probability propagation, where the previous posterior probabilities were multiplied by a transitional probability matrix learned from the data; mode-specific configuration, *n* LDA classifiers, where *n* is the number of steady-state locomotion modes, each trained with data from its corresponding steady-state mode and transitions from that mode to another (all different classes), and where their use was based on the previous detected locomotion mode; and the mode-specific + time-history configuration, a similar architecture as the previous one but with a classifier on each of the gait events where it was applied the Bayes' law and prior probability distribution.

The evaluation of the models used the metric of accuracy percentage but separated transitional error and steady-state error. They were evaluated using user-independent cross-validation, partially-dependent cross-validation and user-dependent validation. For user-independent classification, the pattern recognition system was trained with data from seven of the subjects and classifier performance was tested using data from the eighth subject. This was repeated eight times so that each subject was withheld once (i.e., leave-one-out cross validation). For partially dependent classification, seven trials of steady-state level walking of the novel subject were included in the training pool. User-dependent classification was performed by training the classifier with data from all but one trial for each subject (54 total trials). The withheld trial was used to test the classifier. Cross validation was done across all 54 trials such that each trial was left out once. Best results were always using user-dependent models. The mode-specific configuration yielded the best **transitional error (5% error)** closely followed by the mode-specific + time-history configuration (7% error). In the case of steady-state error the best one **was time history configuration (1% error)** followed by the mode-specific + time-history (2% error) and the mode specific configuration (3% error). On average for both kinds of error, the mode-specific configuration was best.

Woodward et Al. (Woodward, Spanias, & Hargrove, 2016) developed an intention recognition system that used **artificial neural networks** and **LDA classifiers** to recognize locomotion modes (**level walking, stairs ascent, stairs descent and ramp descent**).

Six unilateral lower limb amputees (five male and one female, five transfemoral and one knee disarticulation, four with left and two with right leg amputation, aged between 23

and 65, height between 1.6 and 1.93 m, and weight between 52 and 96 kg) participated in the study.

They were instrumented **with encoders and potentiometers on the knee and ankle joints embedded on the prosthesis**, as well as a **6DOF IMU sensor and 6DOF load cell on the middle of the shank**. These were sampled at 500 Hz. An experimenter manually triggered mode transitions, allowing the amputee to switch seamlessly between locomotion modes.

Subjects completed 20 repetitions of a locomotion circuit that included level ground walking, ramps, and stairs, and necessary transitions between these locomotion modes. For the ramp circuit, each subject walked on level ground, walked up a ramp, transitioned back to level walking for approximately two strides and stopped. The subject turned around and walked back to and down the ramp and continued walking on level ground for a few additional strides before stopping. For the stair circuit, each subject walked on level ground, walked up a four-step staircase with a reciprocal gait transitioned back to level walking for approximately two strides and stopped. The subject turned around and walked back to and down the stairs with a reciprocal gait and continued walking on level ground for a few additional strides before stopping.

The obtained features were the **mean, standard deviation, min, max, initial value and final value of the relative knee and ankle positions, velocities and torques as well as from the 6 axis of the load cell and the 3 axes of the accelerometer and gyroscope from the IMU sensor**. These were retrieved from a 300ms window prior to a toe-off or heel-strike event. The features were labeled with the corresponding locomotion mode (level-walking and ramp ascent had the same controller and were thus included in the same class) as well as the type of step (transitional or steady-state).

Both the ANN and LDA classifier were evaluated using subject-dependent and independent classification where in the subject-independent method a leave-one-out cross-validation using the groups of data for each subject was used and for subject-dependent repeated (10 times) holdout with 25% test data was used. The used metric was accuracy. ANN performed better than LDA both for subject-dependent (**steady-state steps: 0.5% error; transitional step: 6,5% error**) and subject-independent (steady-state steps: 2.5% error; transitional step: 12% error).

Martinez-Hernandez et Al. (Martinez-Hernandez, Mahmood, & Dehghani-Sanij, 2017) developed an algorithm that used a Bayesian formulation for the recognition of locomotion

modes (**level-walking, ramp ascent and ramp descent**) and gait phases (swing and stance phase).

Eight male subjects participated in the study with ages ranged between 24 and 34 years old, heights were between 1.74m and 1.79m, and weights ranged between 77.6 kg and 85 kg.

They were instrumented with I**MU sensors composed of accelerometer and gyroscope on one of the legs attached to the thigh, shank and foot**. These were sampled at 1000 Hz.

Participants were asked to walk at their self-selected speed and complete five repetitions of three locomotion modes: level-ground walking, ramp ascent and ramp descent.

Obtained features were the angular velocities from all IMU sensors. Each sample was labeled with the correct classes for the gait cycle (initial contact, loading response, mid stance, terminal stance, pre-swing, initial swing, mid swing and terminal swing) and locomotion mode. Each gait cycle corresponded to one of two gait phases: stance phase (period 1 to 5) and swing phase (period 6 to 8). Based on these a Bayesian measurement model was built using the data histograms. Samples are iteratively added to an analysis window and analyzed starting from the beginning of the step by the Bayesian model to associate them with the correct gait cycle and locomotion mode by iteratively updating the probability of the current sample belonging to a certain class of gait cycle and locomotion mode.

The model was tested by randomly selecting sensor samples from the testing dataset with 10000 iterations. The metric used was accuracy. In locomotion mode the best result was achieved at **0.13% error** while gait cycle phases were recognized with the lowest error of 0.8% error.

4. METHODOLOGY

This chapter describes the methods and protocols used for the development of the locomotion mode recognition and prediction models for both human subjects and a humanoid robot. This includes a description of the procedures for the data collection, treatment and labeling and for model building and evaluation.

A pipeline for the development of the gait intention classifier was designed to provide a framework for the quick testing of different techniques on many of the separate phases and the creation of the final models. This pipeline runs and is dependent of already implemented machine learning algorithms in MATLAB 2017b. A schematic of the pipeline is represented in Figure 7.

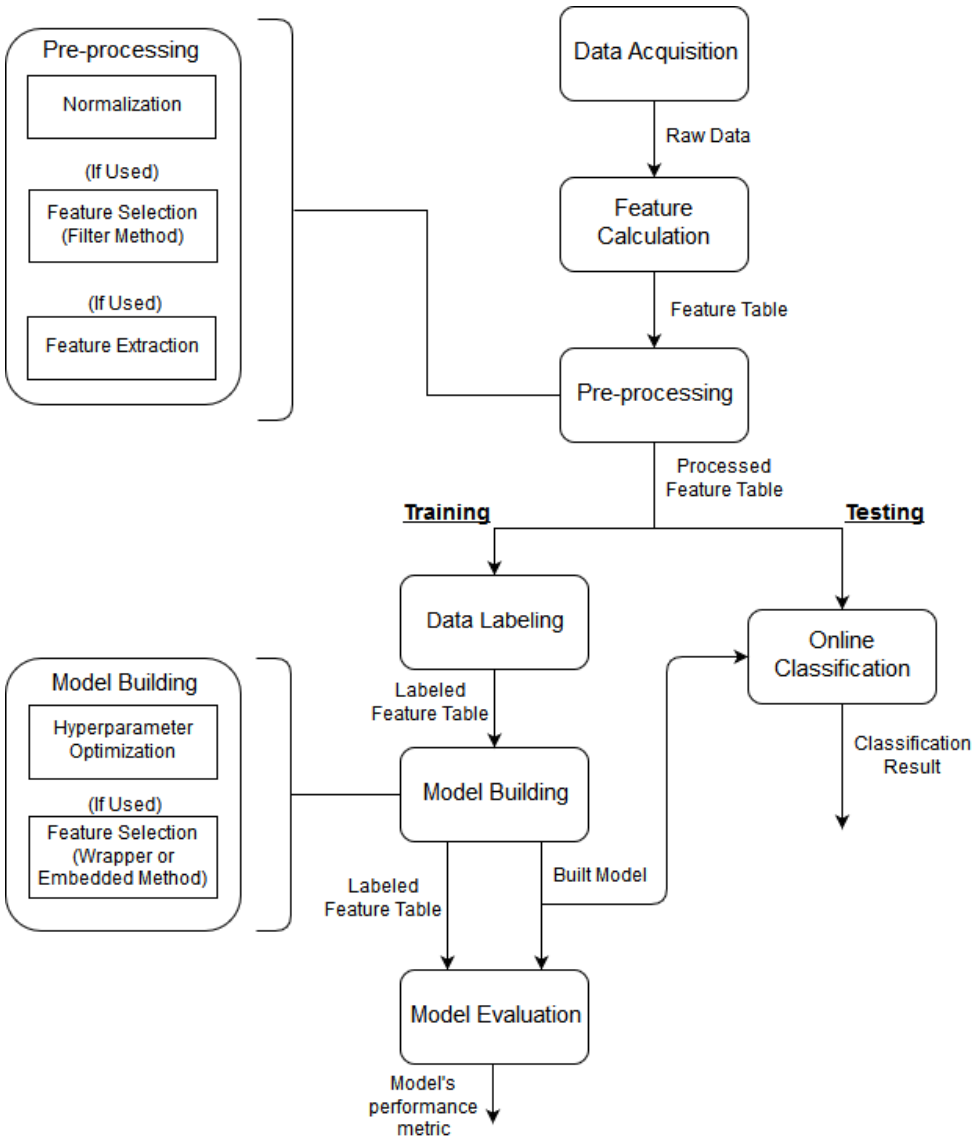

Figure 7 - Pipeline Diagram

4.1 Data Acquisition

To build a useful model, data from known circumstances must be collected and labelled for it to be useful on the training of the model. This data is obtained from two sources: a simulated humanoid robot, and real-world experiments performed with human subjects wearing an array of inertial sensors on the lower back, legs and feet. All experiments are done on several types of terrain such as ramps, level ground and stairs to obtain data from the most common obstacles that human gait must adapt to.

4.1.1 Humanoid Robot

To first validate the usefulness of a locomotion mode classification model and to obtain quickly preliminary data from a controlled environment a simulation of a humanoid robot walking in different environments was set up. Despite this providing a fast and versatile framework for data collection however, it is comprised of ideal scenarios were no external influences affect the robot's gait and where data is not affected by noise, although a certain quantity of it is artificially added, being limited in how well it can mimic normal human gait.

Due to controller limitations, only ramps and level ground were tested with no transitions from two different locomotion modes ever occurring on the same trial. This makes it impossible to test all the different conditions a human subject must deal with on their daily lives.

Using the simulator Webots (version 7.4.3) a robot is made to walk in a generated map. The simulated robot is a DarwIn-OP with a controller based on Central Pattern Generators (CPGs). The robot's gait can be configured to increase or decrease both the speed and the turn angle through the controller parameters.

Simulations are run in a flat ground, ascending or descending slope environment. Several types of ascending and descending slopes were used (2, 5 and 10 degrees of inclination) to simulate real-world existing ramps. Each simulation is run for 30 seconds three consecutive times with the same parameters. Since the robot has some noise added to its gait (2% maximum deviation from the original value) it generates slightly different gaits for the same conditions. Figure 8 shows an example of a simulation being run.

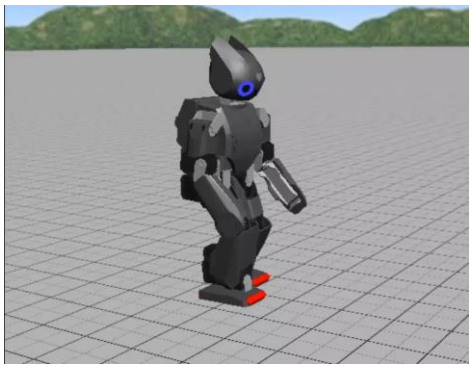

Figure 8 - DarwIn-Op in the simulated environment

Data from the robot is sampled at a frequency of 125 Hz and consists of the pitch, yaw and roll of the left and right hip joint, the pitch of both knee joints and the pitch and yaw of both ankle joints. These angles are marked in Figure 9.

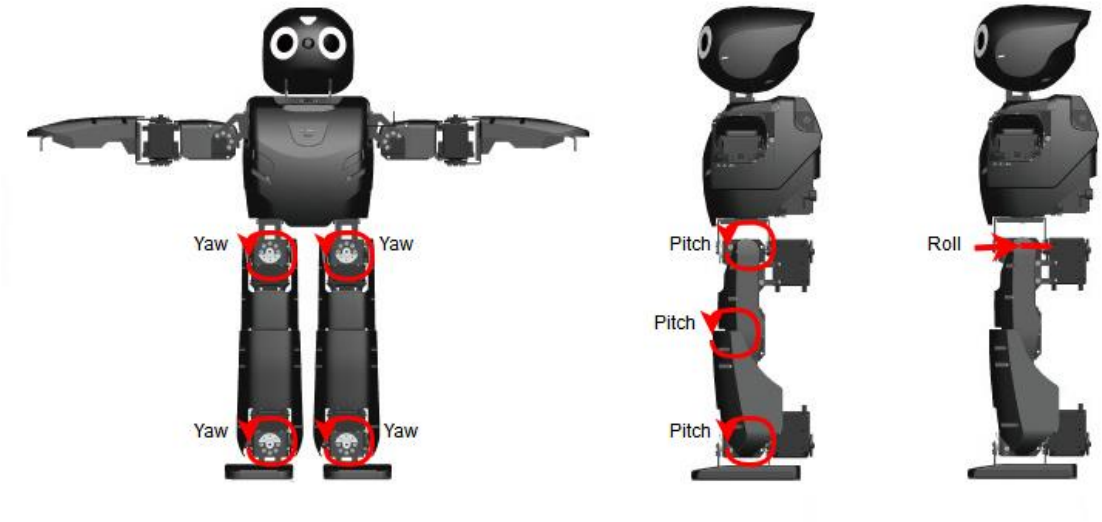

Figure 9 - DarwIn-OP and the obtained joint angles (the red lines indicate the rotation axis)

4.1.2 Human

After the testing of the robot's model and to develop a model capable of being used for locomotion mode classification in real-world situations, data from human subjects was collected.

Human subjects were instrumented with several IMU sensors to gather mechanical data from the body. These kinds of sensors were chosen because of their easy installation and minimal maintenance required. Another option would be EMG sensors, but these are less than ideal because despite being able to provide information that, combined with what is retrieved from the inertial sensors, allows for a more complete picture of the user's gait, they are hard to install and keep attached during the user's daily locomotion.

The IMU sensors are placed on the outer side of the thighs and shanks as well as on top of the feet. One more IMU is placed on the lower back for direction intention estimation (Young, Kuiken, & Hargrove, 2014). The sensors' disposition is shown in Figure 10.

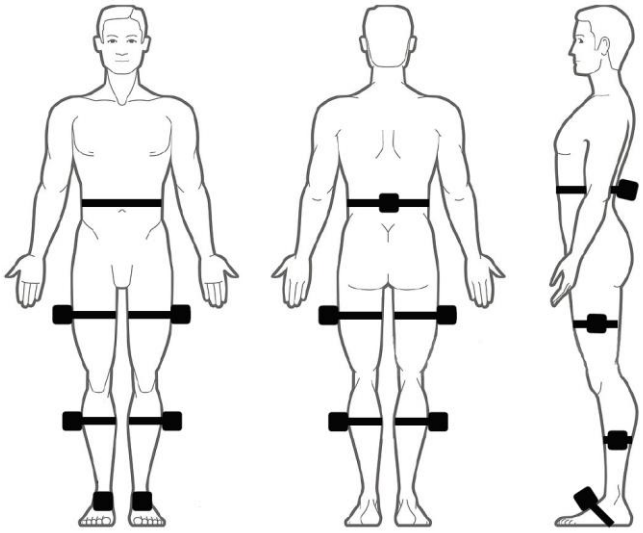

Figure 10 - IMU placement

The sensors are sampled at 200 Hz. The retrieved data was the leg segments' (thigh, shank, foot) angles and angular velocity, torso pitch and rotation angle and torso angular velocity.

As an experimental protocol subjects are asked to walk in different directions and types of terrain. Six subjects (5 male and 1 female with a mean age of 24 ± 1.37 years, mean height of 170 ± 9 cm and mean weight of 63 ± 10.02 Kg), participated in the experimental trials. For each direction, walking forward, backwards, clockwise and anti-clockwise, 9 trials were done with different speeds.

In each of the trials the subjects walked for 10 m and the first three trials of each direction were performed at a fast gait speed, the next three trials at a normal gait speed and

the last three trials at a slow gait speed. They were also asked to perform ten trials in four different circuits at a self-selected gait speed. These circuits are shown in Figure 11.

In the first circuit they would walk 2 meters forward, ascend a staircase, walk forward another two meters and stop, turn around and come back to the starting position. On the second circuit they would have to walk 2 meters forward ascend a ramp and walk forward another two meters, stop, turn around and come back to the starting position. On the two last circuits they would have to walk forward 2 meters, step over an obstacle and walk forward another 2 meters. These were done with two obstacles of different dimensions, one for each circuit. The staircase had 8 steps with 17 cm of height, 31 cm of depth and 110 cm width, the ramp was 10,3 meters with a 10 degrees inclination and the obstacles of the two last circuits were 22 cm in height and 34 cm depth as well as 34 cm in height and 22 cm depth respectively.

Figure 11 - Photos of the circuits where the trials were done

During transitions subjects could lead with any leg. These trials allowed for the collection of data from both level-ground, stairs and ramp steady-state steps, that is, steps that are done on only one type of terrain, and transitional steps from transitions between terrains. An experimenter walked alongside the subjects marking transitional steps. A transitional step was defined as the step when the foot first touched another type of terrain. Some of these trials are displayed in Figure 12.

Figure 12 - Ongoing trials

4.2 Feature Calculation

After the collection of data, it is necessary to obtain a feature vector composed of numerical features from it for the models to classify. These should be meaningful and representative of the data to have the necessary information for a correct classification. They can be obtained from one instance or several instances of data.

This block of the pipeline converts raw data, provided as an input, to a feature vector containing the required information, which is the output.

Using gait events retrieved from the feet gyroscopes' signal, data from a trial is segmented into steps. A feature vector is then calculated from each of the step's signals. A step's boundaries were defined as the Toe-off (TO) events from the same leg when calculating the features for the prediction models and the Heel-strike (HS) events from the same leg when calculating features for recognition models.

Each step's features are calculated from a time window that starts or ends on the time of the event, depending on the model's classification goal and contains data from both legs (Novak et al., 2014). These are stored in a matrix of n x f, of which n is the number of feature vectors or instances and f the number of features.

For recognition purposes, the features are calculated from a time window that starts at the step's HS event. For prediction purposes, the feature values are derived from a time window that ends when the TO event happens. It is helpful to note that prediction features are obtained from a time window prior to the step they will be associated with while recognition

features are acquired from the same step they are associated with. The time window's size is calculated from the elapsed time of the last finished step, making it wider if the step is slower, therefore creating an adaptive window.

The retrieved features were the mean, standard deviation, range and the first and last value from the sampled signals obtained from the adaptive window. Features obtained from human trials are normalized by the subject's height. After its calculation, these features are concatenated into a feature vector.

Features were named according to the joint or segment from which they came from and what the value represented.

Two nomenclatures were tested when naming leg features, one which took into account the side of the leg, where 'left' was used for the left leg and 'right' for the right leg, and another nomenclature that took into account where the event had taken place, where 'event' was used for the leg where the gait event was detected and 'opposite' for the leg opposite to the one where the gait event was detected.

These were followed by the name of the leg joint or segment from which it was calculated, and the specific signal feature calculated, for example, 'left_hip_rot_angle_mean'. The first part of the torso features' names was only named 'torso', for example 'torso_rot_angle_mean'. A detailed explanation of each feature name is depicted on the appendix.

4.3 Pre-processing

The pre-processing stage is important to improve and filter the data on which a model is trained on so that the models' performance is maximized and, in many cases, its' training time reduced. Dissimilar feature value distributions and redundant or incorrect data are also dealt with during this stage. It is also here that some of the techniques of dimensionality reduction are performed.

This block receives as input the unprocessed feature data, applies pre-processing techniques to it and outputs the result. The pre-processing stages used include data normalization, feature selection and feature extraction. During data normalization, features are normalized using one of several different procedures which include centering, standardization, and min-max scaling. This is done to create a common range of the data

where features with larger value range do not reduce the influence of features with smaller ranges. Afterwards either feature selection or feature extraction is performed.

When performing feature selection, filter methods are used to filter noisy and redundant variables to reduce the amount of information required to build a suitable model.

An ANOVA-based method is used on this phase to achieve that goal. This method involves using the mRMR algorithm to rank features in descending order according to their importance. Following that, an ANOVA comparison of the highest ranked feature where the group of feature values associated to each class are compared to assess which classes can be distinguished. This feature is the added to the set of features to be used in the model. If some features are still undistinguishable the same is done on the second-best feature and so on and so forth until there are features that distinguish between all classes.

Feature extraction is usually performed as an alternative to feature selection, since it is done for the same purpose of dimensionality reduction and information filtering. During this phase, the principal component analysis technique is applied to the original feature data in order to extract the most important components. These are selected by applying Horn's Parallel Analysis as a cut-off criteria (Ledesma & Valero-mora, 2007).

4.4 Data Labeling

For supervised classification model to be able to classify data correctly, a preliminary batch of data, labeled in accordance to the event or characteristic associated with each feature vector, is used for the model training. This batch of labeled data becomes the ground truth on which the model bases its decisions for posterior classification. This label takes the form of a numeric or categorical value, called a class, representing different expected output results, meaning that, if the model were to classify that feature vector, the class label is the result that is expected of it.

During this phase, the feature vector data is used to build four databases comprised of the feature vectors and its corresponding label, which are defined as numeric values. Each of these databases is used to train classification models with different purposes. One of them, named '*direction_ft'*, contains every obtained feature vector labeled with the associated direction class. All other databases only contain data that was labeled as data from forward trials, which includes forward level-ground, stairs and ramp trials. The database named '*sts_trs_ft'* contains feature vectors labeled as steady-state or transition step. The remaining

two databases contain a subset of the data from *'sts_trs_ft'*, either the data from steady-state steps, labeled with the type of steady-state step, or the data from transition steps, labeled with the type of transition step. These databases are named *'steady_state_type_ft'* and *'transition_type_ft'* respectively. The databases will later be used to train a set of models for either prediction or recognition.

When features were retrieved from the simulated robot data only two of these databases, '*direction_ft'* and *'steady_state_type_ft'*, are created. This is because since the simulations include no terrain transitions the '*sts_trs_ft'* database would only contain information from steady-state steps and the *'transition_type_ft'* database would contain no information at all.
For the direction database, *'direction_ft'*, the defined classes were 1 for forward direction, 2 for backward direction, 3 for anti-clockwise direction and 4 for clockwise direction.
For the steady-state / transition database, named *'sts_trs_ft'*, the defined classes were 1 for steady-state and 2 for transition.

For *'transition_type_ft'*, the database containing transition step type data, the defined classes were 1 for flat ground to ascending stairs transition, 2 for ascending stairs to flat ground transition, 3 for flat ground to descending stairs transition, 4 for descending stairs to flat ground transition, 5 for flat ground to ascending ramps transition, 6 for ascending ramps to flat ground transition, 7 for flat ground to descending ramps transition, 8 for descending ramps to flat ground transition and 9 for stepping over obstacle transition.
For the database *'steady_state_type_ft'*, containing data of the steady state step type, the classes were defined as 1 for flat ground, 2 for ascending stairs, 3 for descending stairs, 4 for ascending ramp and 5 for descending ramps.

4.5 Model Building

To obtain a usable classification model, it must be built using the assigned training data. This stage can involve optimization of the model's hyperparameters, that is, the parameters that the user defines, as well the application of wrapper or embedded feature selection methods. Using the previously obtained databases, a model is built using each of them. These models will then be used on the next phases. During this phase, hyperparameter optimization is done by building models with every hyperparameter value in a defined interval and comparing the models' performance to find the best one. If feature selection is also to be applied, then it is also done for each parameter value. This phase's feature selection procedure is used, in case no feature selection or extraction was performed in the preprocessing phase, to

construct a feature set containing the features to be used for the model building. This is done by using either “mRMR + Forward Selection” or “Forward Selection + Backwards selection”.

When using “mRMR + Forward Selection” where the features are ranked using the mRMR method and a model is built using the highest ranked feature, i.e. the best, after which the model’s performance is evaluated. Subsequently the next best feature is added to the feature set and the model built and evaluated again. If the feature increases the performance, it is kept, otherwise, it is removed. This is done for every feature or until maximum performance is reached.

When using “Forward Selection + Backwards selection”, the model is trained with every single feature, one at the time. The feature that yields the best performance is then added to the feature set. Afterwards, the model is trained with the selected feature combined with each of the remaining features. The feature that improves the performance the most in combination with the already established feature set is added to the set. This is done until the model performance stops increasing or cannot increase anymore. After that backwards selection is used on the obtained feature set where the process is inverted: features are iteratively removed from the feature set if their absence improves or does not affect performance.

Four different classification algorithms are used in this phase: Discriminant analysis (DA), K-Nearest Neighbors (KNN), Random Forests (RF), Support Vector Machines (SVM). These were chosen due to their prevalence in previous literature (Figueiredo, Santos, Pons, & Moreno, 2016).

Both linear and quadratic discriminant analysis algorithms are tested. K-nearest neighbors classifiers are built using both weighted and unweighted (regular) neighbor distances. Support Vector machines are built using a linear, quadratic, cubic and gaussian kernel.

All models except for discriminant analysis models go through hyperparameter optimization and feature selection procedures.

When using discriminant analysis models, no hyperparameters are tuned and so the default values provided by MATLAB are used. KNN models are tuned by incrementally increasing the number of nearest neighbors (*k)* starting with $k = 1$ until performance reaches the maximum value or starts decreasing.

If random forests are used the number of decision trees, or learners, is tuned. This is done, similarly to the KNN models, by incrementally increasing the number of trees starting with 1 until performance reaches the maximum value or starts decreasing.

When using Support Vector Machines (SVM) models the box constraint hyperparameters C is tuned and if a gaussian kernel is used then the kernel scale hyperparameters, σ, is also tuned. A range of values is defined for the hyperparameters that require tuning. If only C is tuned, then an approach like KNN is used where one starts with the minimum value of the defined range and iteratively uses the next biggest value. In case both hyperparameters need tuning a grid-search is used and every parameter pair is tested with the pair yielding the best performance being chosen. The range picked for both parameters is an interval of exponents where the parameter takes the value 2^y and the value y ranges from [-10: 10] (Duan, Wang, Liu, & Liu, 2010).

After the hyperparameter optimization and/or feature selection the final model is subsequently built with the best parameters and features.

4.6 Model Evaluation

After a model is built cross-validation is used to calculate a performance metric to evaluate the model. This is necessary for the comparison of the resulting models that may have been built using different techniques on any of the pipeline's phases or when comparing models built with different hyperparameters or features. It is also required to evaluate the generalization capability of the model when dealing with previously unseen data.

If the goal is to report the final model's performance, performance is evaluated using repeated 20-fold cross-validation. If it is model selection, that is, when models are being compared, the evaluation is performed by using repeated 2-fold cross-validation. All cross-validation procedures are repeated 10 times. Nested cross-validation is also used. This means that when evaluating the model, its building process is repeated for every fold and the resulting model used to predict the test set from the outer loop. Inner loop cross-validation used for hyperparameter or feature selection is thus the repeated 2-fold cross-validation whereas the outer loop consists of repeated 20-fold cross-validation (Yongli Zhang & Yang, 2015).

For this phase, the MCC metric is used. Specifically, this metric is used for both comparison and reporting of model's performances. This metric was chosen due to its' good

representative properties of unbalanced classes classification results (Jurman, Riccadonna, & Furlanello, 2012).

4.7 Online Classification

Automatic detection of toe-off events is possible using appropriate algorithms on the gyroscope data from the feet thus allowing the calculation of the required feature vector ad consequent classification. For a model to be usable in an online fashion the classification time cannot be superior to the time the classification result must be used at. This is something that must be considered when choosing the model since a late locomotion mode change of the assistive device is done puts the user in risk of falling.

To obtain a complete assessment of the gait conditions from which the data being classified was retrieved, a chain of sequential classification models is used. Each of these models classify the data in a certain way and based on that classification another model is used to obtain a finer classification of the action.

Depending on whether the pipeline is used on robot or human data the training phase outputs two or four models respectively, one for each built database (*'direction_ft'* and *'steady_state_type_ft'* for the robot and *'direction_ft'*, *'sts_trs_ft'*, *'transition_type_ft'* and *'steady_state_type_ft'* for the human). These can be recognition or prediction models depending on the goal.

In case data from the humanoid robot simulated gait is being analyzed then two models are built. The first model classifies the step data according to the walking direction. If the feature vector, is classified as "forward direction", then a steady-state type model is used for the final classification. Both classification results are outputted. This is shown in Figure 13.

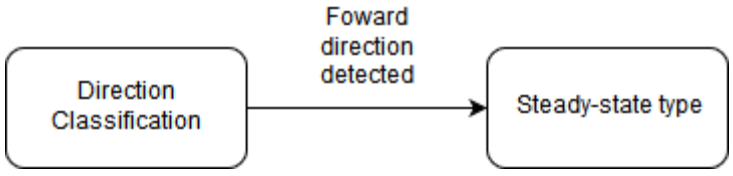

Figure 13 - Schematic of classification models sequence for robot data

When human gait data is used, four models are built. The first model classifies the step data according to the walking direction. If the feature vector, i.e. a step, is classified as

"forward direction", then the step is classified according to whether it is a "steady-state" or a "transition" type. In case a step is classified as steady-state, a steady-state type model is used for the final classification. On the other hand, if a transition is detected then the transition type model is used for the final classification.

After the classification sequence, all results are outputted. A schematic of this sequence is shown in Figure 14.

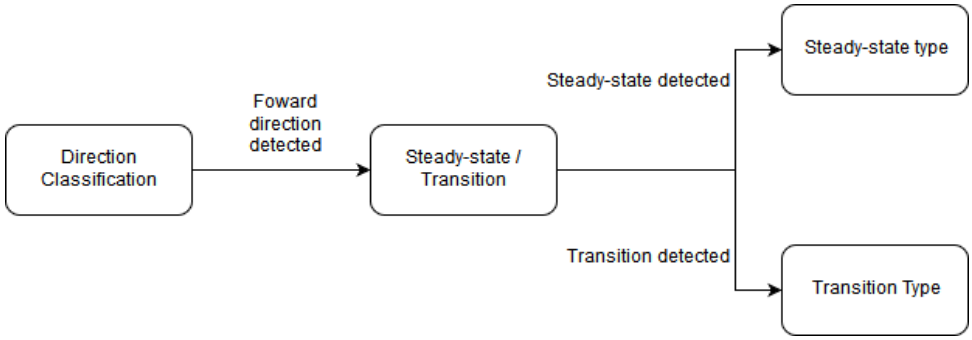

Figure 14 - Schematic of classification models sequence for human data

5. PROCEDURE TESTS

The designed model construction pipeline consists of several phases that are sequentially followed. For some of these phases several techniques can be used to achieve the same goal. Therefore, in this chapter tests are performed for each of these phases to determine the technique, algorithm or parameters that ensure the best performing algorithm. This involved creating models using each of the different techniques, evaluating their performance and comparing them.

The models' evaluation was done using 2-fold repeated cross-validation and the MCC metric was chosen as the representative result for this evaluation.

5.1 Feature Calculation

After the data segmentation into steps, through the usage of both TO and HS events, it was necessary to find which window size would be most representative of the step, and for it to be adaptive window, this size would have to be relative to the step's time length. As such in this chapter the procedure to obtain the size of the analysis window, from which the values of the features to be used in the training of the classification models are to be retrieved, is detailed.

To correctly obtain the desired feature values from the raw data it was necessary to obtain the correct size of the adaptive event window for both prediction, where the window ended at the first TO event of the step, and recognition, where the window started at first HS event of the step.

5.1.1 Methods

To reduce computational time and assuming the window size is common between subjects and robot, the tests were performed using the data set of one random human individual. These tests involved training a KNN model with $k = 1$ using all features, since this kind of models are trained quickly and yield reasonable results. Several different window sizes were tested.

The window size was established as a fraction of the step from which the features are calculated (before the TO event for prediction and after the HS event for recognition).

The features were calculated from several fractions of the step (full-step, half-step, 1/3, 1/4, 1/5 and 1/6), as well two different feature nomenclatures (features were named in relation to either the left or right leg or in relation to the event leg or the opposite leg) for both prediction and recognition.

The models' performance was assessed using leave-one-out cross-validation which outputted the value of the MCC metric.

5.1.2 Results

The average performance results of all recognition models (direction, steady-state/transition, transition type and steady-state type) for both feature nomenclatures is detailed in Figure 15. The results are separated by the fraction of the step from which the features were calculated. These show a clear advantage when calculating features from a time window that encompasses the full step.

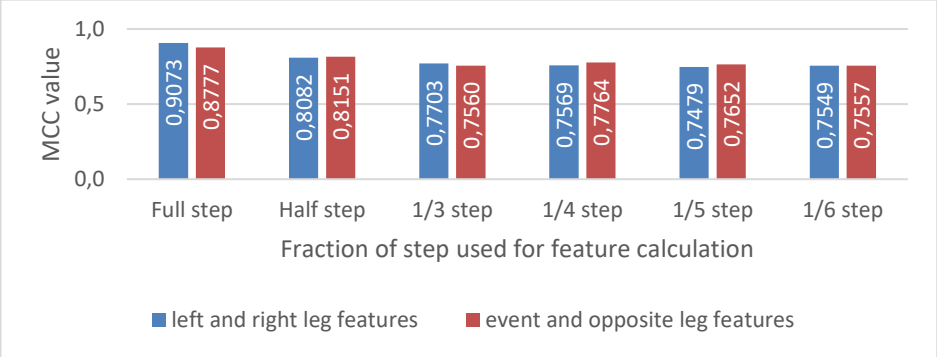

Figure 15 – Average Recognition Model Performance

Figure 16 16 represents the same results but for the prediction models. Despite the performance values being somewhat similar it shows that calculating features from 1/4 of the last step yields the better results.

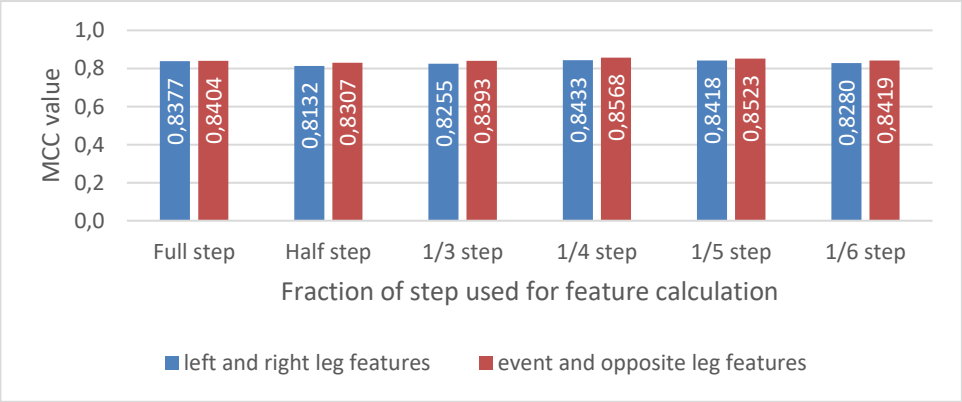

Figure 16 – Average Prediction Model Performance

5.1.3 Discussion

Since these tests only involve the calculation of features through basic mathematical operations none of these windows affect computation time when calculating the features. Results of the locomotion mode recognition models show that using the full step for classification outperforms all the other cases by a significant margin. Using the nomenclature of left and right leg features seems to have a positive effect on the performance as well. It is notable that, generally, the less of a step was used for recognition the worse the results were.

For prediction the results are different. Using 1/4 of the step yields the best results and, contrary to the recognition case, using the event and opposite leg feature nomenclature has the best performance. Using half of the step yielded the worst results, however, changing the step size mostly has a small effect on the model's performance.

Following these results on the remainder of the procedures described in this document, features will be calculated, for recognition purposes, from the full step with the left and right leg nomenclature and, for prediction purposes, features will be calculated from 1/4 of the step with the event and opposite leg nomenclature.

5.2 Normalization

Normalization is a key step in data processing, that can diminish the time required for a classification model to be trained and increase its performance. There are however several different techniques which depending on the data and the classification algorithm can have a better or worse effect on the classification outcome. Because of this, tests must be done to find out the proper technique to be used on the building of the final models. This chapter details these tests, performed with some of the most common normalization procedures and their effect on the models' performance.

5.2.1 Methods

To analyze the effects of data normalization on the studied classifier algorithms these were trained and evaluated with prediction and recognition data from all subjects normalized using 4 different techniques. All features were used, and no model tuning was performed,

meaning the default parameter values provided by MATLAB were used. The studied normalization techniques were centering, z-score standardizing and min-max scaling with both [0; 1] and [-1; 1] intervals. The results were also compared to models built with data that was not normalized.

5.2.2 Results

Figure 17 reports the average results across all databases (direction, steady-state / transition, transition type and steady-state type) and models. They are presented for both prediction and recognition models.

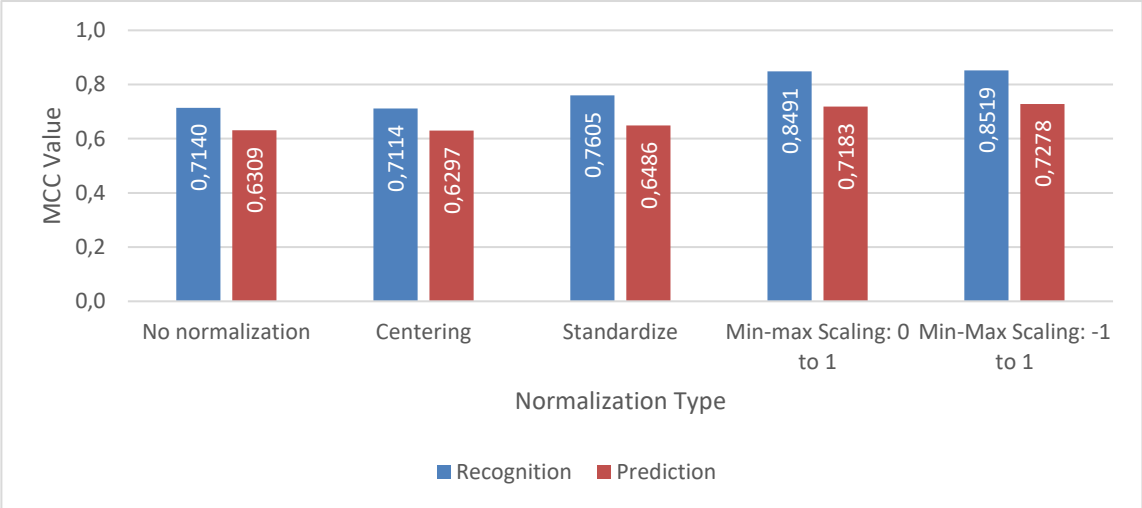

Figure 17 - Average MCC value of the classification model per normalization technique

To allow for a more complete analysis of the normalization techniques, results for each of the used classification algorithms are also reported.

Figure 18 and Figure 19 show the performance results for the discriminant analysis models averaged across all databases.

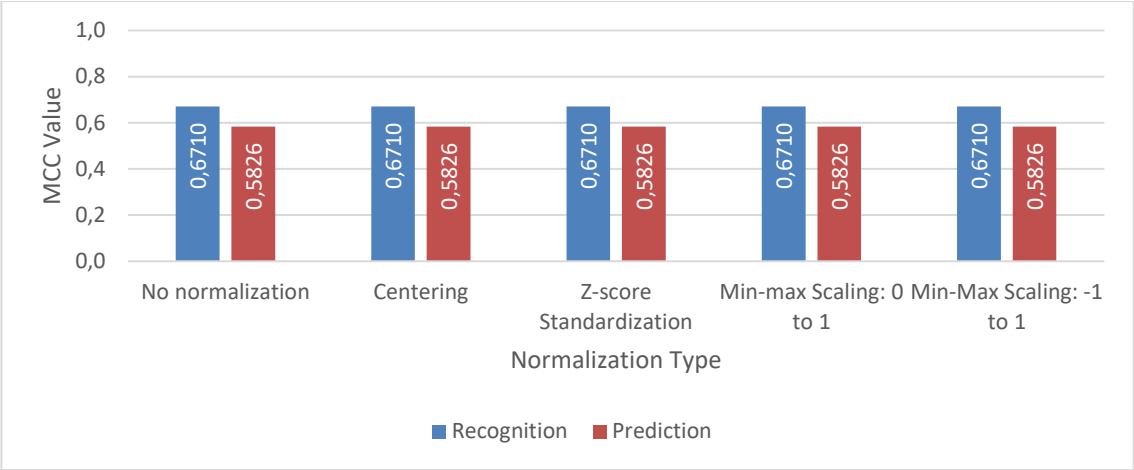

Figure 18 - Average MCC value of the Linear DA models by normalization type

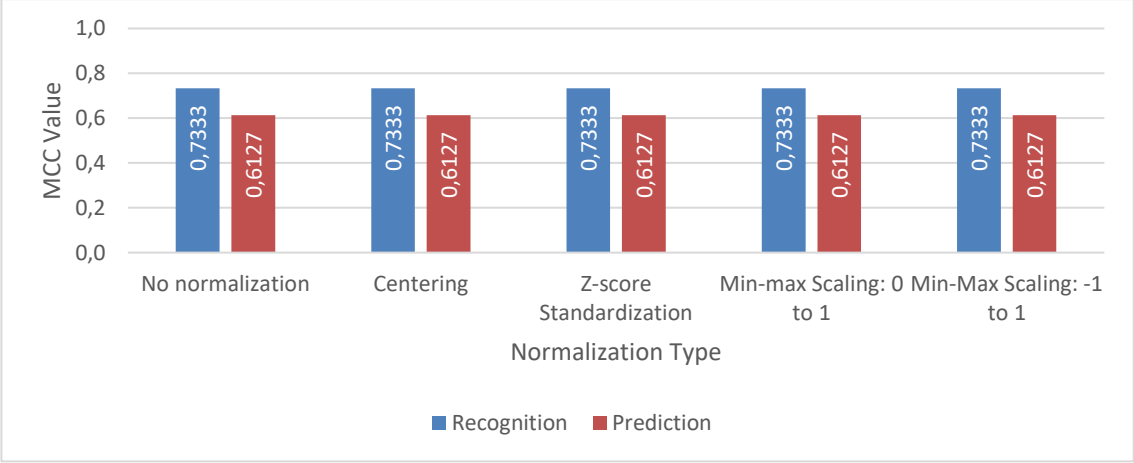

Figure 19 - Average MCC value of the Quadratic DA models by normalization type

Figure 20 and Figure 21 show the performance results of the KNN models averaged across all databases.

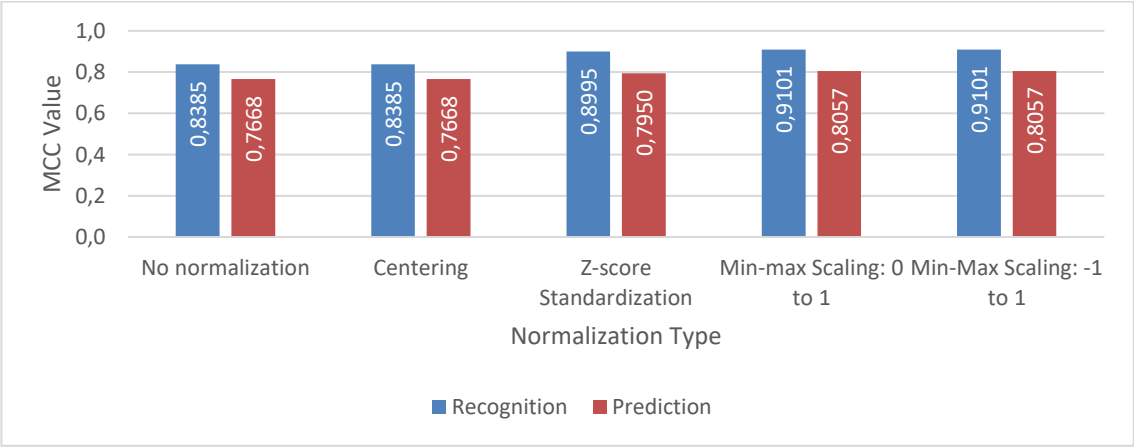

Figure 20 - Average MCC value of the Regular KNN models by normalization type

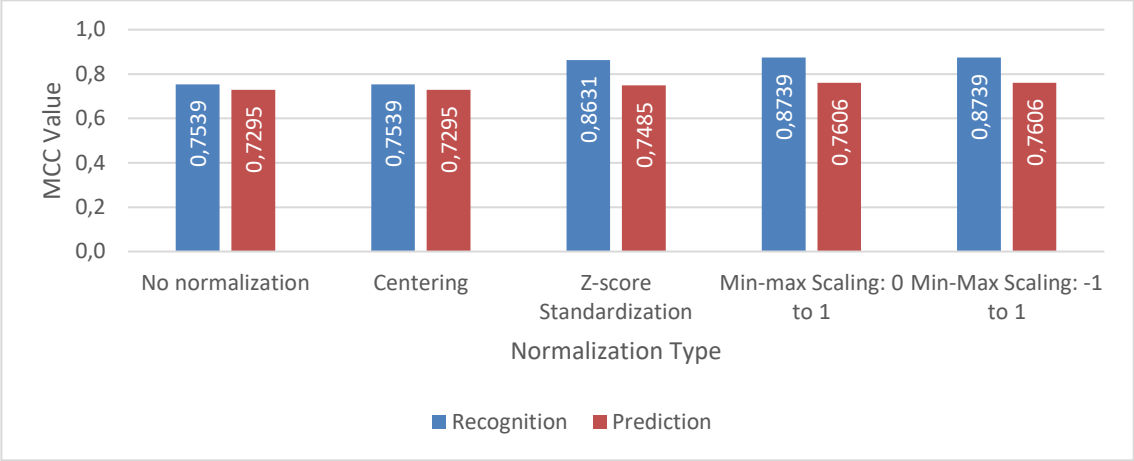

Figure 21 - Average MCC value of the Weighted KNN models by normalization type

Figure 22 shows the performance results of the Random Forests models averaged across all databases.

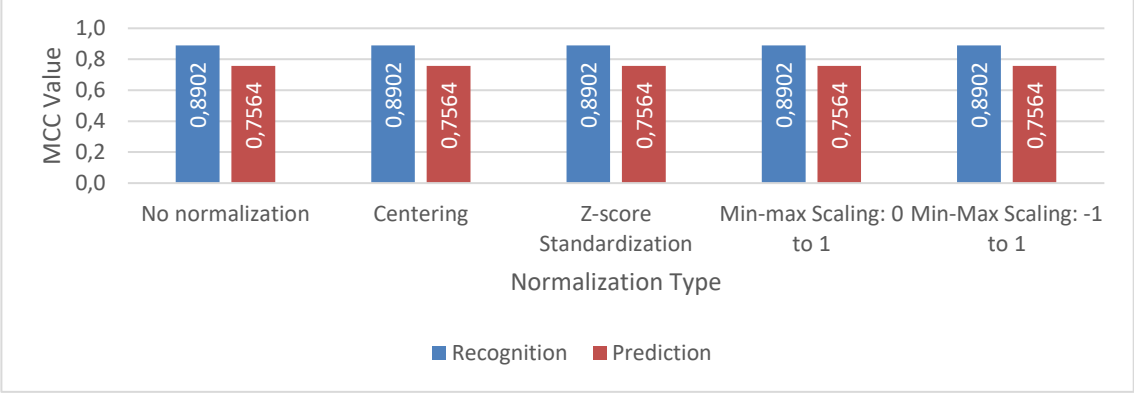

Figure 22 - Average MCC value of the Random Forests models by normalization type

Figure 23, Figure 24, Figure 25 and Figure 26 display the performance results for the Support Vector Machines (SVM) models averaged across all databases.

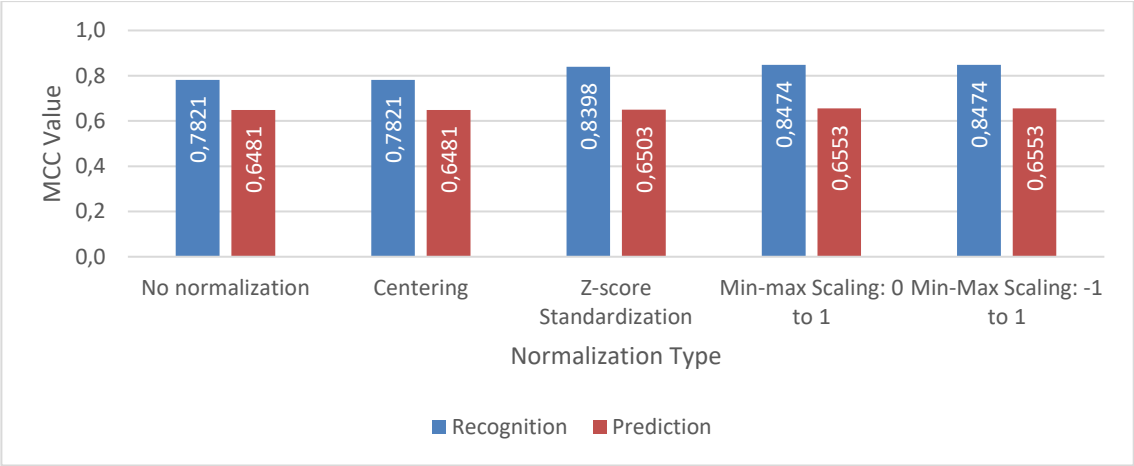

Figure 23 - Average MCC value of the Linear SVM models by normalization type

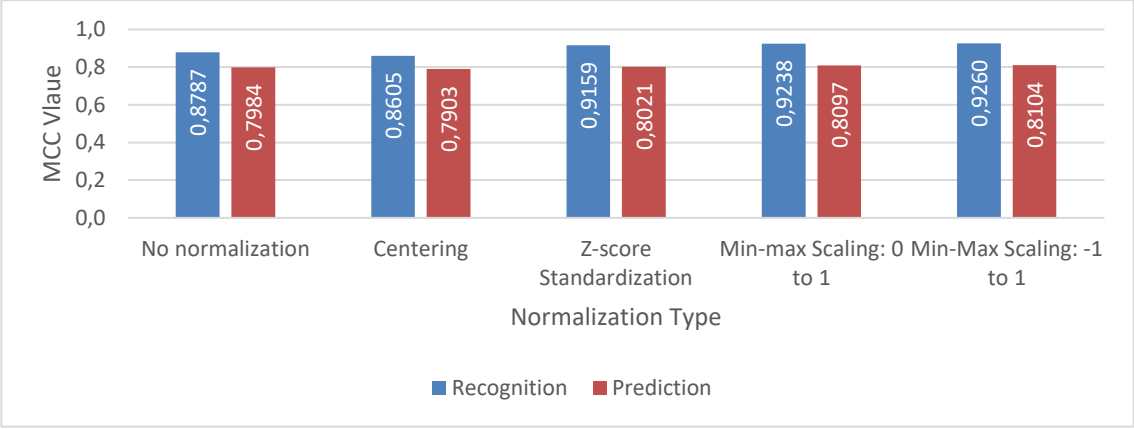

Figure 24 - Average MCC value of the Quadratic SVM models by normalization type

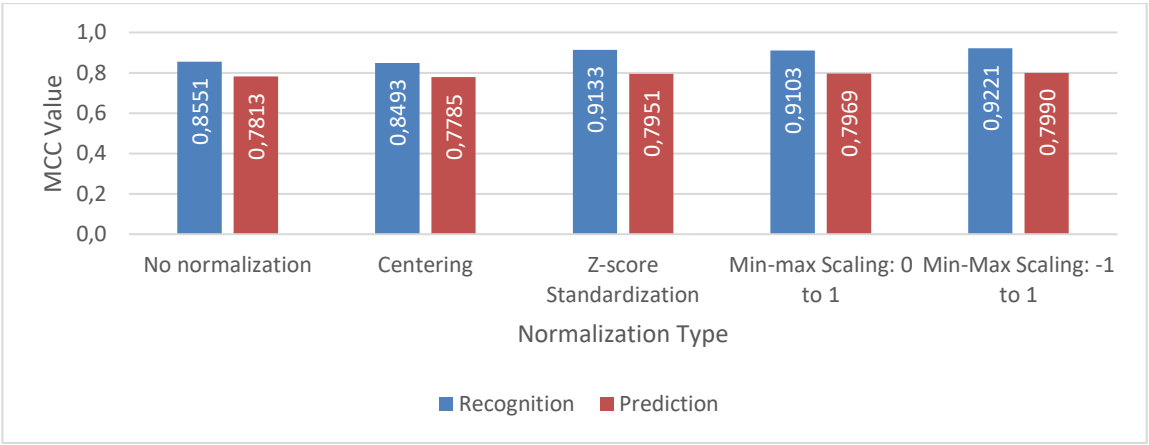

Figure 25 - Average Performance of the Cubic SVM models by normalization type

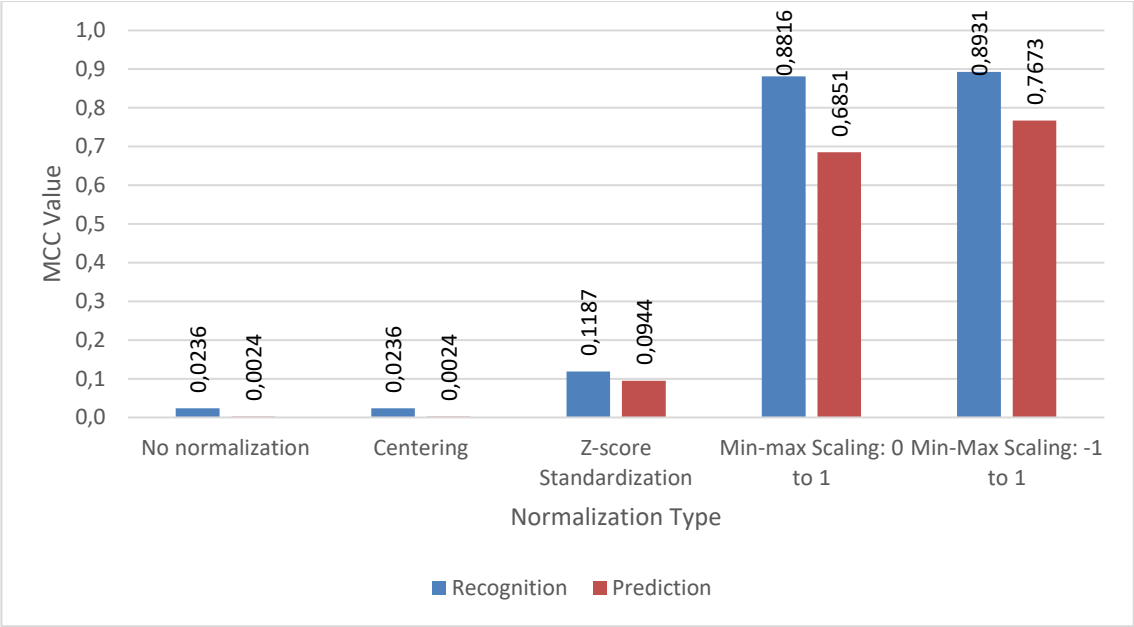

Figure 26 - Average MCC value of the Gaussian SVM models by normalization type

5.2.3 Discussion

In earlier research it was very common to use the z-score standardization technique (Begg & Kamruzzaman, 2005; Hanson, Jr., Barth, Lach, & Brandt-Pearce, 2009; Wu, Wang, & Liu, 2007) while some use min-max scaling with an interval of [0; 1] (Novak et al., 2013). However, the normalization technique used in these kinds of studies is not referenced very often and sometimes not even used making it difficult for a comparison between the results obtained in this document and since the normalization effect and technique is very dependent on the data and goal there are no studies providing a one-fits-all solution to the problem.

Regarding the obtained results, it is notable that normalization had no effect on discriminant analysis, possibly since these models work by reorienting the data in accordance to the features' axis and thus scaling has no effect. Random forest classifiers were also unaffected.

Using no normalization or centering the data had the same effect on all classifiers except for linear, quadratic and cubic SVM. Normalization had the most dramatic effect on the gaussian SVM models, where results were very poor for all techniques except for the min-max scaling ones.

Using min-max scaling with the interval [-1; 1] yielded the best results for both recognition and prediction and thus was the technique chosen for the remainder of the tests described in this document.

5.3 Feature Selection and Extraction

When dealing with a large number of features and in order to distill the important information from the data feature selection and extraction algorithms are used. Besides removing superfluous features these can also increase a model's performance and classification speed by removing noisy and misleading ones. There are however many ways of performing this dimensionality reduction and while there may not be a perfect algorithm some different techniques may be tested to achieve reasonable results.

In this chapter the results of tests that evaluated the performance impact of several feature extraction and selection techniques on the classification models are reported. These were done to determine the best technique to use in the model building pipeline.

5.3.1 Methods

Tests were performed with a k-nearest neighbor (KNN) classifier with $k = 1$ using the recognition and prediction data from a random human subject, and normalizing the data using the min-max scaling with the interval [-1; 1] as selected in the previous chapter. Only a KNN classifier was used in order to obtain results in an acceptable frame of time, since these are trained quickly and yield good results.

Four different procedures were studied that included Principal component analysis, a feature extraction procedure, and for feature selection an ANOVA-based procedure, a forward selection procedure combined with a mRMR feature ranking and a combination of regular feature selection followed by backward selection.

5.3.2 Results

Figure 27 reports the average performance of both the recognition and the prediction models after using each of the different feature selection and extraction techniques.

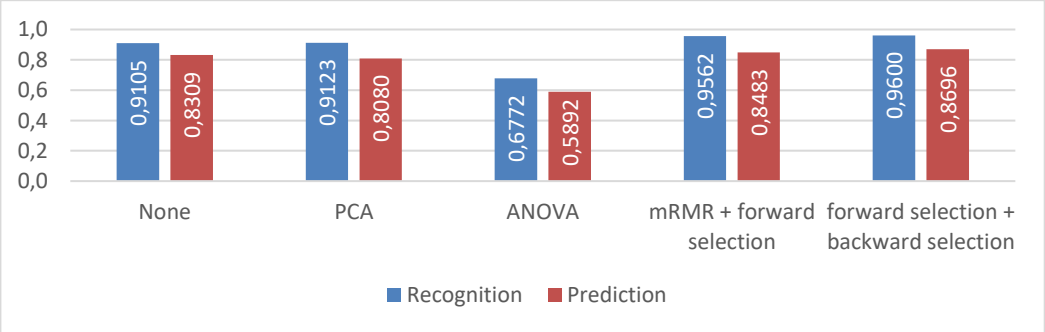

Figure 27 - Average Performance per feature selection technique

Figure 28 reports the average number of features selected by each procedure.

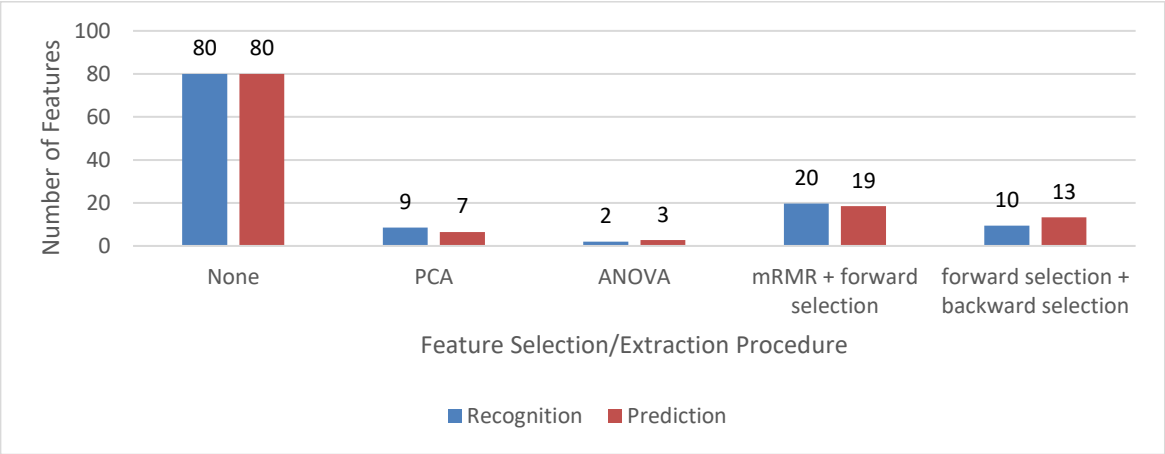

Figure 28 - Number of selected features by feature selection technique

5.3.3 Discussion

Using the mRMR + forward selection method or the combination of forward and backward selection yields similar results, but the former is less computationally intensive and so, despite it picking a larger number of features than the combination of forward and backward selection, it is the selected method for feature selection during the remainder of this document. These findings are consistent with the earlier literature (F. Zhang & Huang, 2013) where the mRMR method was compared to the forward selection and backward selection separately and a conclusion was reached that mRMR was faster and yielded similar results.

These sequential selection and ranking-based techniques tend to be the most commonly used procedures in previous studies (Chan et al., 2014; Leuenberger et al., 2014; F. Zhang & Huang, 2013).

Using the ANOVA method however was yielded the worst results due to the number of features that were chosen. This is, due to the way the algorithm is implemented, because it only selects a maximum of as many features as the number of classes to identify. Since one single feature is not enough to discern between the classes the results are poor.

5.4 Classifier Algorithm Selection

There are several various kinds of classification algorithms that were developed for application on the field of machine learning. Some, currently dubbed “traditional machine learning” algorithms, include k-nearest-neighbors, Support vector machines, decision trees and others. All these algorithms work under different concepts and assumptions and as such have their own uses, advantages and disadvantages.

This chapter reports the results of tests performed with these algorithms to find out which one was better suited for the creation of the locomotion mode prediction and recognition models.

5.4.1 Methods

Using the prediction and recognition data of all subjects and the min-max scaling method with an interval of [-1; 1] for normalization, previously discovered as the best one, as well as using all features, several kinds of machine learning algorithms, namely linear and quadratic discriminant analysis models, nearest-neighbors models with both weighted and unweighted (regular) distances, random forests and support vector machines with a linear, polynomial and gaussian kernel were tested.

The model parameters were also tuned using an iterative approach, that is, testing each value one by one, to achieve the best possible performance value for each model to have a fair comparison between them.

5.4.2 Results

Figure 29 shows the results of the tests for the recognition and prediction models built with each of the classification algorithms averaged across every database (direction, steady-state/transition, steady-state type and transition type).

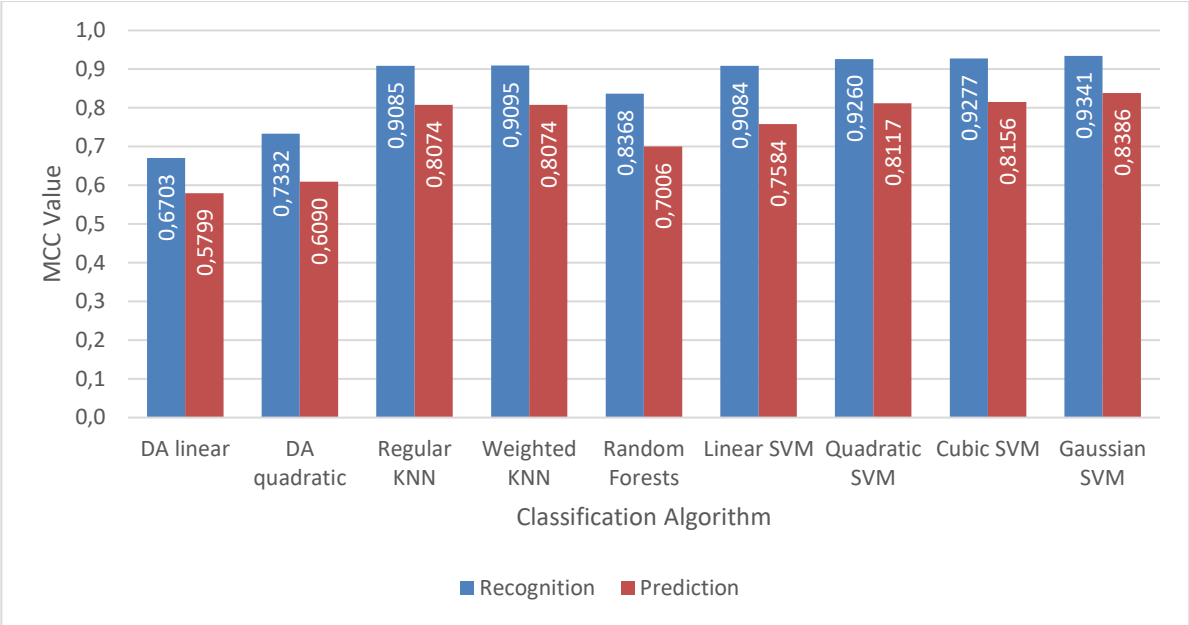

Figure 29 - Average MCC value of the models for each classification algorithm

Figure 30 shows the average amount of time models obtained from each classification algorithm take to classify a feature vector.

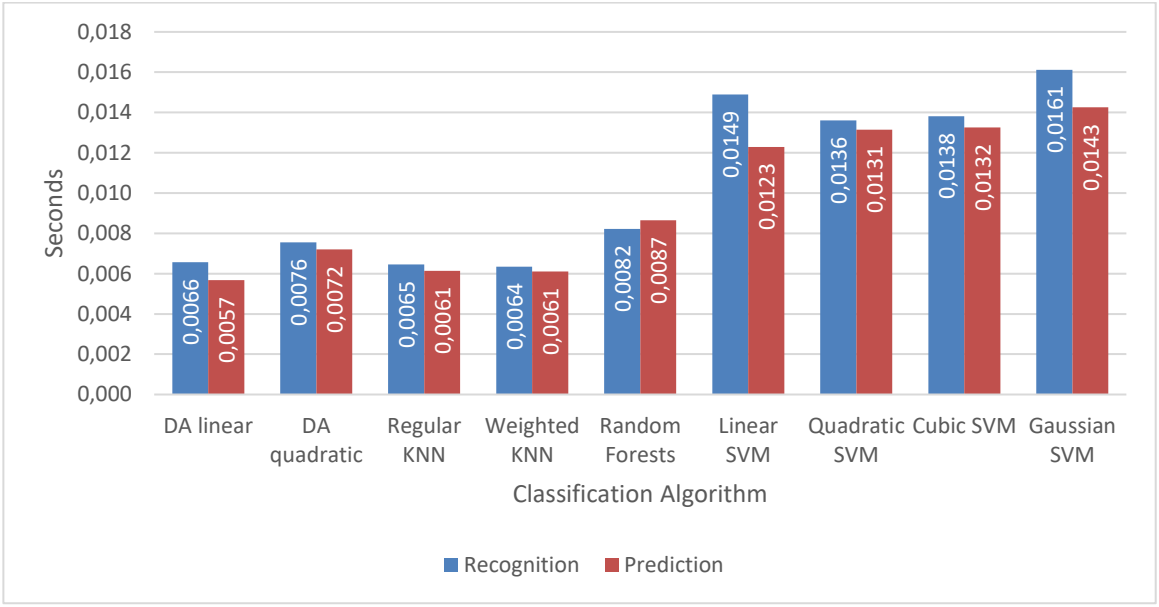

Figure 30 - Average classification time for each classification algorithm

5.4.3 Discussion

Results show that the Support vector machine classifier with a gaussian kernel performed better than other algorithms for both prediction and recognition and as such it will be the one used for building the final models. This is likely because, by having two

parameters instead of one, this model inherently has more flexibility to suit the data it was provided with than other models. Previous literature reports also consistently show this algorithm as the better one (Huang et al., 2011; Leuenberger et al., 2014; F. Zhang & Huang, 2013). However, due to the increased model complexity, this can also lead to a higher chance of overfitting and, like with other kinds of support vector machines algorithms, the resulting model takes twice as long to classify data as other algorithms.

Both linear discriminant analysis models had the worst results most likely due to their simplicity.

The KNN models take less time to classify data and their performance results are similar to the gaussian SVM and as such demonstrate potential, after further future scrutiny, to possibly become a better alternative. It is notable that both types of KNN models achieved very similar performance since the number of used nearest neighbors was also very similar (1 for the regular KNN models and 2 or 3 for the weighted KNN models) and as such both types of models' decisions were based on similar data.

6. LOCOMOTION MODE RECOGNITION

One of the goals of the designed pipeline was the building of locomotion mode recognition models. After performing tests to evaluate which techniques were to be used on each of the phases of the pipeline, this chapter documents the performance results for the obtained final recognition models to assess its robustness and fitness of use.

The models were evaluated using 20-fold repeated cross-validation and the MCC metric was chosen as the representative result for this evaluation. The evaluation's confusion matrix is also shown for a more in-depth analysis.

6.1 Methods

Using the techniques that were previously found to be the best for each of the pipeline's phases, the final recognition models were built and tested. As such features were calculated from each full step with a left/right leg nomenclature, min-max scaling with an interval of [-1; 1] was used for data normalization, the mRMR + forward selection algorithm for feature selection and the gaussian SVM algorithm for model building.

These models were built from the humanoid robot data, resulting in two models (direction classification and steady-state type classification) and the human gait data resulting in four models (direction classification, steady-state/transition classification, transition step type and steady state step type classification).

The human gait recognition models were built using data from all subjects.

6.2 Results

To properly evaluate the recognition models for both cases (robot and human) results were separated into their own sections. Each of these sections document the MCC value for each of the models and its corresponding confusion matrix for both the humanoid robot and human subjects.

The number of steps analyzed is defined by the total number of predictions and the analysis of each row of the confusion matrix allows for an assessment of which classes the model struggled to classify.

6.2.1 Humanoid Robot

Table 1 represents the evaluation results for the recognition models using robot gait data. These were the direction classification model (*'direction_ft'*) and steady-state step type classification (*'steady_state_type_ft'*).

Table 1 – Results of the evaluation of the robot gait models

Model	Number of Classified Steps	Number of used features	MCC value
direction_ft	39279	30	1
steady_state_type_ft	12287	3	1

Table 2 and Table 3 represent the confusion matrices of the classification results showing the number of correct and incorrect predictions for each of the classes.

Table 2 - Confusion matrix of recognition model trained with database 'direction_ft'

Predicted class / True class	Forward	Backward	Counter-clockwise	Clockwise
Forward	12287	0	0	0
Backward	0	12530	0	0
Counter-clockwise	0	0	7924	0
Clockwise	0	0	0	6538

Table 3 - Confusion matrix of recognition model trained with database 'steady_state_type_ft'

Predicted class / True class	Level-walking	Ascending ramp	Descending ramp
Level-walking	2492	0	0
Ascending ramp	0	5018	0
Descending ramp	0	0	4777

Features that were selected for all robot recognition models are listed in Table 4. A list of the features selected for each model is detailed in the appendix.

Table 4 - Selected features common to all robot gait recognition models

Feature	Segment / Joint
Maximum value of the tilt angle signal	Left ankle joint
Minimum value of the tilt angle signal	
Minimum value of the tilt angle signal	Right ankle joint

6.2.2 Human

Table 5 summarizes the evaluation results for the recognition models using human gait data. These were the direction classification model (*'direction_ft'*), steady-state/transition classification model (*'sts_trs_ft'*), transition step type classification model (*'transition_type_ft'*) and steady-state step type classification model (*'steady_state_type_ft'*).

Table 5 - Results of the evaluation of the human gait classification models

Model	Number of Classified Steps	Number of used features	MCC value
direction_ft	6064	43	0.9979
sts_trs_ft	3170	69	0.8166
transition_type_ft	300	19	0.9928
steady_state_type_ft	2870	53	0.9945

Table 6, Table 7,

Table 8 and Table 9 represent the confusion matrices of the classification results showing the number of correct and incorrect predictions for each of the classes to allow for a more in-depth analysis of the results.

Table 6 - Confusion matrix of the recognition model trained with database 'direction_ft'

Predicted class / **True class**	Forward	Backward	Counter-clockwise	Clockwise
Forward	3167	2	1	3
Backward	0	907	0	0
Counter-clockwise	2	0	1024	0
Clockwise	1	0	0	957

Table 7 - Confusion matrix of the recognition model trained with database 'sts_trs_ft'

Predicted class / **True class**	Steady-state	Transition
Steady-state	2859	83

Transition	11	217

Table 8 - Confusion matrix of the recognition model trained with database 'transition_type_ft' (LW – level-walking; AS – ascending stairs; DS – descending stairs; AR – ascending ramps; DR – descending ramps; SO – stepping over obstacle)

Predicted class / True class	LW to AS	AS to LW	LW to DS	DS to LW	LW to AR	AR to LW	LW to DR	DR to LW	SO
LW to AS	30	0	0	0	0	0	0	0	0
AS to LW	0	30	0	0	0	0	0	0	0
LW to DS	0	0	30	0	0	0	0	0	0
DS to LW	0	0	0	30	0	0	0	0	0
LW to AR	0	0	0	0	30	1	0	0	0
AR to LW	0	0	0	0	0	29	0	0	0
LW to DR	0	0	0	0	0	0	30	0	1
DR to LW	0	0	0	0	0	0	0	30	0
SO	0	0	0	0	0	0	0	0	59

Table 9 - Confusion matrix of the recognition model trained with database 'steady_state_type_ft'

Predicted class / True class	Level-walking	Ascending stairs	Descending stairs	Ascending ramp	Descending ramp
Level-walking	1634	0	1	1	5
Ascending stairs	0	210	0	0	0
Descending stairs	0	0	209	0	0
Ascending ramp	1	0	0	386	0
Descending ramp	2	0	0	0	421

Since the number of features used by some of the models was high, a complete list of the selected features for each model is present in the appendix. Features that were selected for all human recognition models are listed in Table 10.

Table 10 - Selected features common to all human gait recognition models

Feature	**Segment / Joint**
Last value of the tilt angle signal	Left thigh
Mean value of the tilt angular velocity signal	
First value of the tilt angle signal	Left shank
Standard Deviation of the tilt angle signal	Left foot
Standard deviation of the tilt angular velocity signal	

Range of the tilt angular velocity signal	
Mean value of the tilt angular velocity signal	Right thigh
Mean value of the tilt angular velocity signal	Right shank

6.3 Discussion

The recognition models built from the robot's data show flawless performance, having classified every step correctly with a relatively small number of features. 30 features were used for the direction classification model and 3 for the steady-state type classification model. These results are very promising and show that the models may also have a satisfactory performance on humans.

It is worth noting though that these results are to be expected due to the lower amount classes that the models had to discern from, at least in the steady state type classification case. Another reason would be the lower variability of data that is obtained from a simulated environment when compared to a real one. All these factors make these models less representative of the human data-based models than what would be ideal.

Models obtained by training with human data show poorer results than the robot models, even if some of the models show a performance very close to perfect. This is most likely due to the higher variability in data since it was obtained from several subjects and the lower number of steps that were used to train the models.

The direction classification model obtained from human data had near-perfect results (MCC = 0.9979) with only a few forward steps being classified as counter-clockwise or clockwise and vice-versa. This was achieved by using 43 features, slightly more than half of the original features showing that not all information was required for a correct assessment of the step's direction.

The steady-state / transition classification model, only used when building the models with human data, had the worst results (MCC = 0.8166) even when using the most information (69 features). This might be either because of errors in the obtained data or because the used features were not enough to successfully distinguish between a transitional or a steady-state step.

The transition type classification model, also only used when building the models from human data, had very good results (MCC = 0.9928) with errors when classifying transitions

from level walking to ramps. The model was built with 19 features, the least number of features out of all four of the human gait classification ones. This shows that distinguishing transition step types are possible with a high accuracy and with a small amount of information, as long as the step is identified as a transitional one.

The steady-state type classification type, similarly to all models but the steady-state/transition classification one, had near-perfect results (MCC = 0.9945). Errors were due to the classification of level walking steps as ramp steps and vice-versa. This model was built from 53 features.

In previous literature (Tkach & Hargrove, 2013; Young & Hargrove, 2016) it was reported that the inclusion of ramps as one of the studied locomotion modes introduced some error due to the similarities between ramps and level-ground walking. This corresponds with the obtained results, where all errors involved the walking on or transitioning to ramps, thus explaining why these kinds of errors prevailed the most, and also why the steady-state / transition classification model showed such poor results since all ramp and level walking trials were grouped into two categories making the errors more pronounced.

Other than the steady-state/transition model, however, the evaluation results were higher or at least in line with what was reported by the aforementioned studies, even though different metrics were used, as those also revealed high classification accuracy.

7. LOCOMOTION MODE PREDICTION

The second goal of the developed pipeline was the creation of robust locomotion mode prediction models and, like in the previous chapter, these were built using the best techniques and algorithms as per the previously done procedure tests. This chapter summarizes all the performance results for the final prediction models and the features they used.

The models were evaluated using 20-fold repeated cross-validation and the MCC metric was chosen as the representative result for this evaluation. The evaluation's confusion matrix is also shown for a more in-depth analysis.

7.1 Methods

The models were trained using features calculated from data pertaining to a quarter of the steps that preceded the toe-off events of the step to be predicted. Features were named and sorted according to the event/opposite nomenclature. For data normalization, min-max scaling with an interval of [-1; 1] was used and for feature selection the mRMR + forward selection algorithm. Finally, the models were built using the gaussian SVM algorithm.

Using the humanoid robot data, resulted in two models (direction classification and steady-state type classification) while the human gait data resulted in four models (direction classification, steady-state/transition classification, transition step type and steady state step type classification).

The human gait prediction models were built using data from all subjects.

7.2 Results

The results are separated into two sections corresponding to the humanoid robot models and the human models respectively. In these sections the MCC value of each of the built models are documented as well as their confusion matrices. The number of features each model uses and which ones are common to them are also documented.

The number of steps analyzed is defined by the total number of predictions and the analysis of each row of the confusion matrix allows for an assessment of which classes the model struggled to classify.

7.2.1 Humanoid Robot

Table 11 represents the evaluation results for the prediction models using robot gait data. These were the direction classification model (*'direction_ft'*) and steady-state step type classification (*'steady_state_type_ft'*).

Table 11 – Results of the evaluation of the robot gait classification models

Model	Number of Classified Steps	Number of used features	MCC value
direction_ft	39279	15	1
steady_state_type_ft	12287	4	1

Table 12 and Table 13 represent the confusion matrices of the classification results showing the number of correct and incorrect predictions for each of the classes.

Table 12 - Confusion matrix of the recognition model trained with database 'direction_ft'

Predicted class / True class	Forward	Backward	Counter-clockwise	Clockwise
Forward	12287	0	0	0
Backward	0	12530	0	0
Counter-clockwise	0	0	7924	0
Clockwise	0	0	0	6538

Table 13 - Confusion matrix of recognition model trained with database 'steady_state_type_ft'

Predicted class / True class	Level-walking	Ascending ramp	Descending ramp
Level-walking	2492	0	0
Ascending ramp	0	5018	0
Descending ramp	0	0	4777

Since the number of features used by some of the models was high, a complete list of the selected features for each model is present in the appendix. Features that were selected for all human recognition models are listed in Table 14.

Table 14 - Selected features common to all robot gait prediction models

Feature	Segment / Joint
Last value of the tilt angle signal	Event ankle joint
Max value of the roll angle signal	Opposite hip joint
Range of the pitch angle signal	Opposite ankle joint

7.2.2 Human

Table 15 represents the evaluation results for the prediction models using human gait data. These were the direction classification model (*'direction_ft'*), steady-state/transition classification (*'sts_trs*_ft'), transition step type (*'transition_type_ft')* and steady-state step type classification (*'steady_state_type'*).

Table 15 – Results of the evaluation of the human gait classification models

Database	Number of Classified Steps	Number of used features	MCC value
direction_ft	6070	52	0.9897
sts_trs_ft	3176	64	0.6065
transition_type_ft	302	38	0.8866
steady_state_type_ft	2866	59	0.9857

Table 16, Table 17, Table 18 and Table 19 represent the confusion matrices of the classification results showing the number of correct and incorrect predictions for each of the classes to allow for a more in-depth analysis of the results.

Table 16 - Confusion matrix of the recognition model trained with database 'direction_ft'

Predicted class / True class	Forward	Backward	Counter-clockwise	Clockwise
Forward	3167	0	19	12
Backward	0	909	0	0
Counter-clockwise	5	0	1006	0
Clockwise	4	0	0	948

Table 17 - Confusion matrix of the recognition model trained with database 'sts_trs_ft'

Predicted class / True class	Steady-state	Transition
Steady-state	2867	173
Transition	9	127

Table 18 - Confusion matrix of the recognition model trained with database 'transition_type_ft' (LW – level-walking; AS – ascending stairs; DS – descending stairs; AR – ascending ramps; DR – descending ramps; SO – stepping over obstacle)

Predicted class / True class	LW to AS	AS to LW	LW to DS	DS to LW	LW to AR	AR to LW	LW to DR	DR to LW	SO

LW to AS	26	0	5	0	1	1	0	0	1
AS to LW	0	29	0	0	0	0	0	0	0
LW to DS	2	0	24	2	0	0	0	0	3
DS to LW	0	0	0	28	0	0	0	0	0
LW to AR	0	0	0	0	27	0	0	2	0
AR to LW	0	1	0	0	0	28	0	0	0
LW to DR	1	0	0	0	2	1	28	0	1
DR to LW	0	0	0	0	0	0	0	27	0
SO	2	0	2	0	0	0	2	1	55

Table 19 - Confusion matrix of the recognition model trained with database 'steady_state_type_ft'

Predicted class / **True class**	Level-walking	Ascending stairs	Descending stairs	Ascending ramp	Descending ramp
Level-walking	1630	0	0	4	8
Ascending stairs	1	209	0	0	0
Descending stairs	0	0	210	0	0
Ascending ramp	1	0	0	383	0
Descending ramp	2	0	0	0	418

Since the number of features used by some of the models was high, a complete list of the selected features for each model is present in the appendix. Features that were selected for all human recognition models are listed in Table 20.

Table 20 - Selected features common to all human gait prediction models

Feature	**Segment / Joint**
Standard Deviation of the tilt angular velocity signal	Event thigh
Standard Deviation of the tilt angle signal	Event shank
First value of the tilt angular velocity signal	
Last value of the tilt angular velocity signal	
Mean value of the tilt angle signal	Event foot
Standard deviation of the tilt angle signal	
Range of the tilt angle signal	
First value of the tilt angle signal	
First value of the tilt angle signal	Opposite thigh
First value of the tilt angle signal	
Mean value of the tilt angle signal	Opposite shank
Last value of the tilt angle signal	
First value of the tilt angular velocity signal	
Mean value of the tilt angle signal	Opposite foot
First value of the tilt angle signal	
Mean value of the tilt angular velocity signal	
Last value of the tilt angle signal	
Mean value of the tilt angular velocity signal	Torso

7.3 Discussion

Regarding the models obtained from the humanoid robot data, results were perfect with no classification errors. This, just like with the recognition models, has to do with the low variability inherent to the robot gait and the fact that there were less labels to be classified.

The number of features used for each model was also low (15 features for the direction classification model and 4 features for the steady-state type classification model).

However, and also like the recognition models, the lower number of terrain types and the lack of transitions make these models less representative of the human gait-based ones than desirable.

The models obtained from human gait data display poorer results, although some of the models have a performance close to perfect.

One of these near-perfect models is the direction classification model (MCC = 0.9897) with only a few forward steps being classified as counter-clockwise or clockwise and vice-versa, similarly to the human recognition models. The model used 52 features, slightly more then half of the total number of features, showing that there were still quite a few features irrelevant to the model. Novak (Novak et al., 2014), in his automatic turn detection system reported results similar to the ones reported by the modes used here (97% to 99%).

The steady-state / transition classification model, also like its analogous recognition model, had the worst results (MCC = 0.6065) and used the most information (64 features).

This is most likely because the mechanical features obtained before a step is made, like the ones used in to train the models, are not enough to predict that step's type.

The transition type classification model had poor results as well (MCC = 0.8866) with errors when predicting most transitions except for ascending stairs to level walking, descending stairs to level walking and descending ramps to level walking. 38 features were used. These results reinforce the idea that there is not enough information for the model to be able to correctly predict transitions.

The steady-state type classification model is among the better ones with an MCC value of 0.9857, using 59 features. However, it must be noted that this might not mean that the model can correctly predict a locomotion mode most of the time since every step that followed the classified data was of the same class, making the classification biased. This shows that a steady-state classification model might not be needed since the locomotion mode

remains the same until a transition occurs. This means it is more useful to only predict if a transition will occur as well as what kind of transition.

Most previous literature (Chan et al., 2014; Huang et al., 2011; Liu et al., 2016; Woodward et al., 2016; F. Zhang & Huang, 2013) did not define transitional steps as their own class, rather, a boundary was defined between locomotion modes after which the next locomotion mode was attributed. These reported 0 missed transitions and a timely detection of said transitions showing these approaches were better than the one tested in this document. The fact that information from EMG sensors from these sensors and that subjects were prosthesis users, who have different gait patterns from healthy subjects, namely starting transitioning with the instrumented leg (prosthesis) after the healthy leg has already transitioned, might have had a positive impact on the results and explain the disparity with what was observed here.

Similar studies to this one (Tkach & Hargrove, 2013; Young & Hargrove, 2016), that used transitions as a different class also reported lower errors on transitional stages (12% and 5% respectively) while using only mechanical sensors such as the ones used in this study. This can also be due to the aforementioned different gait patterns exhibited by prosthesis users.

8. CONCLUSIONS

Several techniques and classification algorithms were tested in order to obtain an adequate pipeline for the creation of classification models for locomotion mode recognition and prediction. The results of these experiments are reported in this chapter as well as considerations about the work done and notes about what to test and implement in future work.

8.1 Final Considerations

To obtain well-performing gait recognition and prediction machine learning algorithms several techniques were compared across all stages of the data collection and model creation. This led to the design of the best possible pipeline for the creation of locomotion mode recognition and prediction models.

When it comes to feature calculation, obtaining data from the full step with a left/right leg feature nomenclature for recognition purposes and from ¼ of the step with an event/opposite leg feature nomenclature for prediction purposes yielded the best results.

In the normalization stage, using the min-max scaling with an interval of [-1; 1] was the selected method.

For dimensionality reduction, a combination of the mRMR and the forward selection algorithm was selected as the used procedure, however, the regular forward selection method yielded the same results for less features but took longer to run.

The selected classification algorithm for building the final models was the gaussian SVM.

These models performed well for both prediction and recognition of the robot gait, however these are not analogous to the human models since no mode transitions were considered. For the human setting the models revealed themselves to be good for the prediction and recognition of the gait direction but incapable of accurately distinguishing and predicting transitional steps from steady-state steps. Because of this, even if the models' performance for the transition step type and steady-state step type were high, the models failed to achieve the objective: their use on assistive devices for their seamless daily use.

Selected features were high in number and no pattern was discerned from them as most of the features were selected at least once for the models.

Another conclusion is that recognition models might be unnecessary since if a classification model can accurately detect or predict transitions then these can be used to deduct the current locomotion mode. It was also concluded that the use of biomechanical features, especially if the used angle values are less than reliable, is not enough for the proposed goal. It is also necessary to note that different assistive devices and locomotion capabilities of the users warrant different strategies. If, for example, a user has both legs impaired and a rehabilitation using an orthosis on both legs is required then a visual sensor is required to obtain information from the environment in order to change the assistive device's mode.

RQ1: What techniques are most commonly used in previous literature?

Previous literature tested SVM, ANN and KNN classification models as well thresholds or this goal

RQ2: What is the most reliable metric to evaluate the obtained models?

Due to the unbalanced nature of the data when it comes to class distribution, the MCC (Matthews correlation coefficients) is used.

RQ3: Which normalization technique is better for this kind of information and the selected model?

Min-max scaling with an interval [-1; 1] was selected as the best normalization technique.

RQ4: Which is the feature selection procedure that strikes the best balance between computation time and model performance for locomotion mode recognition and prediction?

mRMR ranking + forward selection procedure has the best balance between computational time and model performance.

RQ5: Which classification algorithm is best for the recognition and prediction of locomotion modes?

The gaussian SVM algorithm was considered the best for the building of the models

RQ6: Which features have the most significant role for each classification objective?

Every feature was selected at least once for the models and as such, also due to the high number of selected features, it is not possible to discern a pattern of features.

RQ7: Is it possible to predict user locomotion intentions using only biomechanical data?
It was concluded that the use of only biomechanical data is insufficient for the prediction of user intentions.

8.2 Future work

In future work, the development of a system that incorporates visual feedback and other sensory information, like EMG sensors, and adapts to a user's degree of mobility should be developed. It is also required to improve the robustness of the data acquisition system as well as perform a new study on the required information. Using another approach to transitions such as the one used by previous literature (transition boundary) should also be studied and compared. The possible simplification of the model hierarchy through the removal of the steady-state recognition and prediction models should be tested. Retrieving data from more subjects should also be done to assess if it increases the models' performance.

Finally, the testing of the models in real-time, while implemented on an assistive device such as an orthosis, should also be done.

REFERENCES

Alpaydin, E. (2010). *Introduction to Machine Learning* (2nd ed.). The MIT Press.

Altman, N. S. (1992). An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. *The American Statistician*, *46*(3), 175–185. https://doi.org/10.1080/00031305.1992.10475879

Begg, R., & Kamruzzaman, J. (2005). A machine learning approach for automated recognition of movement patterns using basic, kinetic and kinematic gait data. *Journal of Biomechanics*, *38*(3), 401–408. https://doi.org/https://doi.org/10.1016/j.jbiomech.2004.05.002

Cawley, G. C., & Talbot, N. L. C. (2010). On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation. *Journal of Machine Learning Research*, *11*, 2079−2107. Retrieved from http://jmlr.csail.mit.edu/papers/v11/cawley10a.html%5Cnhttp://www.jmlr.org/papers/volume11/cawley10a/cawley10a.pdf

Chan, H., Yang, M., Wang, H., Zheng, H., McClean, S., Sterritt, R., & Mayagoitia, R. E. (2014). Assessing Gait Patterns of Healthy Adults Climbing Stairs Employing Machine Learning Techniques. *International Journal of Intelligent Systems*, *29*(2), 495–524. https://doi.org/10.1002/int

Chen, B., Zheng, E., & Wang, Q. (2014). A locomotion intent prediction system based on multi-sensor fusion. *Sensors (Switzerland)*, *14*(7), 12349–12369. https://doi.org/10.3390/s140712349

Denisko, D., & Hoffman, M. M. (2018). Classification and interaction in random forests. *Proceedings of the National Academy of Sciences*, *115*(8), 1690 LP-1692. Retrieved from http://www.pnas.org/content/115/8/1690.abstract

Duan, H., Wang, R., Liu, X., & Liu, H. (2010). A method to determine the hyper-parameter range for tuning RBF support vector machines. *2010 International Conference on E-Product E-Service and E-Entertainment, ICEEE2010*, 1–4. https://doi.org/10.1109/ICEEE.2010.5661082

Figueiredo, J., Santos, C. P., Pons, J. L., & Moreno, J. C. (2016). Recognition of Gait Patterns in Human Motor Disorders using Machine Learning: A Review, 1–21.

Guyon, I., & Elisseeff, A. (2003). An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research (JMLR)*, *3*(3), 1157–1182. https://doi.org/10.1016/j.aca.2011.07.027

Haghighat, M., Abdel-Mottaleb, M., & Alhalabi, W. (2016). Discriminant Correlation Analysis: Real-Time Feature Level Fusion for Multimodal Biometric Recognition. *IEEE Transactions on Information Forensics and Security*, *11*(9), 1984–1996. https://doi.org/10.1109/TIFS.2016.2569061

Hanson, M. A., Jr., H. C. P., Barth, A. T., Lach, J., & Brandt-Pearce, M. (2009). Neural Network Gait Classification for On-Body Inertial Sensors. In *2009 Sixth International Workshop on Wearable and Implantable Body Sensor Networks* (pp. 181–186). https://doi.org/10.1109/BSN.2009.48

Hill, T., & Lewicki, P. (2006). *Statistics: Methods and Applications: A Comprehensive Reference for Science, Industry, and Data Mining*. https://doi.org/10.1016/B978-0-323-03707-5.50024-3

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2008). A Practical Guide to Support Vector Classification. *BJU International*, *101*(1), 1396–1400. https://doi.org/10.1177/02632760022050997

Huang, H., Zhang, F., Hargrove, L. J., Dou, Z., Rogers, D. R., & Englehart, K. B. (2011). Continuous Locomotion-Mode Identification for Prosthetic Legs Based on Neuromuscular–Mechanical Fusion, *58*(10), 2867–2875. https://doi.org/10.1109/TBME.2011.2161671

Huihua, Z., Reher, J., Horn, J., Paredes, V., & Ames, A. D. (2015). Realization of stair ascent and motion transitions on prostheses utilizing optimization-based control and intent recognition. *IEEE International Conference on Rehabilitation Robotics*, *2015–Septe*, 265–270. https://doi.org/10.1109/ICORR.2015.7281210

Jang, J., Kim, K., Lee, J., Lim, B., & Shim, Y. (2015). Online gait task recognition algorithm for hip exoskeleton. *IEEE International Conference on Intelligent Robots and Systems*, *2015–Decem*, 5327–5332. https://doi.org/10.1109/IROS.2015.7354129

Jung, J. Y., Chae, M. G., Jang, I. H., & Park, H. (2012). A hybrid control method of an exoskeleton robot for intention-driven walking rehabilitation of stroke patients. *2012 9th International Conference on Ubiquitous Robots and Ambient Intelligence, URAI 2012*, (Urai), 58–60. https://doi.org/10.1109/URAI.2012.6462930

Jurman, G., Riccadonna, S., & Furlanello, C. (2012). A comparison of MCC and CEN error measures in multi-class prediction. *PLoS ONE*, *7*(8), 1–8. https://doi.org/10.1371/journal.pone.0041882

Kohavi, R. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection 2 Methods for Accuracy Estimation. *Proc. of IJCAI'95*, *5*, 1137–1145. https://doi.org/10.1067/mod.2000.109031

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, *97*(1–2), 273–324. https://doi.org/10.1016/S0004-3702(97)00043-X

Kotsiantis, S. B., Kanellopoulos, D., & Pintelas, P. E. (2006). Data preprocessing for supervised learning. *International Journal of Computer Science*, *1*(2), 111–117. https://doi.org/10.1080/02331931003692557

Ledesma, R. D., & Valero-mora, P. (2007). Determining the Number of Factors to Retain in EFA: an easy-to-use computer program for carrying out Parallel Analysis. *Practical Assessment, Research & Evaluation*, *12*(2), 2–11. https://doi.org/http://pareonline.net/getvn.asp?v=12&n=2

Leuenberger, K., Gonzenbach, R., Wiedmer, E., Luft, A., & Gassert, R. (2014). Classification of stair ascent and descent in stroke patients. *Proceedings - 11th International Conference on Wearable and Implantable Body Sensor Networks Workshops, BSN Workshops 2014*, 11–16. https://doi.org/10.1109/BSN.Workshops.2014.10

Li, Y. D., & Hsiao-Wecksler, E. T. (2013). Gait Mode Recognition and Control for a Portable - Powered Ankle - Foot Orthosis. https://doi.org/10.1109/ICORR.2013.6650373

Liu, M., Wang, D., & Helen Huang, H. (2016). Development of an Environment-Aware Locomotion Mode Recognition System for Powered Lower Limb Prostheses. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *24*(4), 434–443. https://doi.org/10.1109/TNSRE.2015.2420539

Lobo, J. M., Jiménez-valverde, A., & Real, R. (2008). AUC: A misleading measure of the performance of predictive distribution models. *Global Ecology and Biogeography*, *17*(2), 145–151. https://doi.org/10.1111/j.1466-8238.2007.00358.x

Martinez-Hernandez, U., Mahmood, I., & Dehghani-Sanij, A. A. (2017). Simultaneous Bayesian recognition of locomotion and gait phases with wearable sensors. *IEEE Sensors Journal*, *18*(3), 1282–1290. https://doi.org/10.1109/JSEN.2017.2782181

Ngo, T. T., Makihara, Y., Nagahara, H., Mukaigawa, Y., & Yagi, Y. (2015). Similar gait action recognition using an inertial sensor. *Pattern Recognition*, *48*(4), 1285–1297. https://doi.org/10.1016/j.patcog.2014.10.012

Novak, D., Goršič, M., Podobnik, J., & Munih, M. (2014). Toward Real-Time Automated

Detection of Turns during Gait Using Wearable Inertial Measurement Units. *Sensors*, *14*(10), 18800–18822. https://doi.org/10.3390/s141018800

Novak, D., Reberšek, P., De Rossi, S. M. M., Donati, M., Podobnik, J., Beravs, T., … Munih, M. (2013). Automated detection of gait initiation and termination using wearable sensors. *Medical Engineering and Physics*, *35*(12), 1713–1720. https://doi.org/10.1016/j.medengphy.2013.07.003

Plotnik, M., Bartsch, R. P., Zeev, A., Giladi, N., & Hausdorff, J. M. (2013). Effects of walking speed on asymmetry and bilateral coordination of gait, *31*(9), 1713–1723. https://doi.org/10.1109/TMI.2012.2196707.Separate

Russell, S. J., & Norvig, P. (2010). *Artificial Intelligence - A Modern Approach*.

Samuel, A. L. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, *3*(3), 210–229. https://doi.org/10.1147/rd.33.0210

SAS. (1989). Introduction : The Basics of Principal Component Analysis. *SAS / STAT Guide*, 1–56.

Tkach, D. C., & Hargrove, L. J. (2013). Neuromechanical sensor fusion yields highest accuracies in predicting ambulation mode transitions for trans-tibial amputees. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 3074–3077. https://doi.org/10.1109/EMBC.2013.6610190

Tucker, M. R., Olivier, J., Pagel, A., Bleuler, H., Bouri, M., Lambercy, O., … Gassert, R. (2015). Control strategies for active lower extremity prosthetics and orthotics: a review. *Journal of NeuroEngineering and Rehabilitation*, *12*(1), 1. https://doi.org/10.1186/1743-0003-12-1

Turker, K. S. (1993). Electromyography: Some methodological problems and issues. *Physical Therapy*, *73*(10), 698–710. https://doi.org/10.1093/ptj/73.10.698

Upton, G., & Cook, I. (2014). *A Dictionary of Statistics*. Retrieved from http://www.oxfordreference.com/view/10.1093/acref/9780199679188.001.0001/acref-9780199679188

Wilson, B. (2002). Cognitive rehabilitation, an integrative neuropsychological approach. *Journal of Neurology, Neurosurgery, and Psychiatry*, *72*(3), 421. https://doi.org/10.1136/jnnp.72.3.421-a

Woodward, R. B., Spanias, J. A., & Hargrove, L. J. (2016). User intent prediction with a scaled conjugate gradient trained artificial neural network for lower limb amputees using a powered prosthesis. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, *2016–Octob*, 6405–6408. https://doi.org/10.1109/EMBC.2016.7592194

Wu, J., Wang, J., & Liu, L. (2007). Feature extraction via KPCA for classification of gait patterns. *Human Movement Science*, *26*(3), 393–411. https://doi.org/https://doi.org/10.1016/j.humov.2007.01.015

Young, A. J., & Hargrove, L. J. (2016). A Classification Method for User-Independent Intent Recognition for Transfemoral Amputees Using Powered Lower Limb Prostheses. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *24*(2), 217–225. https://doi.org/10.1109/TNSRE.2015.2412461

Young, A. J., Kuiken, T. A., & Hargrove, L. J. (2014). Analysis of using EMG and mechanical sensors to enhance intent recognition in powered lower limb prostheses. *Journal of Neural Engineering*, *11*(5), 056021. https://doi.org/10.1088/1741-2560/11/5/056021

Zhang, F., & Huang, H. (2013). Source selection for real-time user intent recognition toward volitional control of artificial legs. *IEEE Journal of Biomedical and Health Informatics*,

17(5), 907–914. https://doi.org/10.1109/JBHI.2012.2236563
Zhang, Y., Ding, C., & Li, T. (2008). Gene selection algorithm by combining reliefF and mRMR. *BMC Genomics*, *9 Suppl 2*, S27. https://doi.org/10.1186/1471-2164-9-S2-S27
Zhang, Y., & Yang, Y. (2015). Cross-Validation for Selecting a Model Selection Procedure, 1–52.
Zheng, H., Yang, M., Wang, H., & Mcclean, S. (2009). Machine Learning and Statistical Approaches to Support the Discrimination of Neuro-degenerative Diseases Based, 57–70.

APPENDIX I

The following tables explain the full meaning of each feature name for both robot and human.

Humanoid Robot:

hip_rot_angle_mean	Mean of the rotation angle signal of the hip joint
hip_rot_angle_std	Standard deviation of the rotation angle signal of the hip joint
hip_rot_angle_max	Maximum value of the rotation angle signal of the hip joint
hip_rot_angle_min	Minimum value of the rotation angle signal of the hip joint
hip_rot_angle_range	Range of the rotation angle signal of the hip joint
hip_rot_angle_first	First value of the rotation angle signal of the hip joint
hip_rot_angle_last	Last value of the rotation angle signal of the hip joint
hip_yaw_angle_mean	Mean value of the yaw angle signal of the hip joint
hip_yaw_angle_std	Standard deviation of the yaw angle signal of the hip joint
hip_yaw_angle_max	Maximum value of the yaw angle signal of the hip joint
hip_yaw_angle_min	Minimum value of the yaw angle signal of the hip joint
hip_yaw_angle_range	Range of the yaw angle signal of the hip joint
hip_yaw_angle_first	First value of the yaw angle signal of the hip joint
hip_yaw_angle_last	Last value of the yaw angle signal of the hip joint
hip_tilt_angle_mean	Mean value of the tilt angle signal of the hip joint
hip_tilt_angle_std	Standard deviation of the tilt angle signal of the hip joint
hip_tilt_angle_max	Maximum value of the tilt angle signal of the hip joint
hip_tilt_angle_min	Minimum value of the tilt angle signal of the hip joint
hip_tilt_angle_range	Range of the tilt angle signal of the hip joint
hip_tilt_angle_first	First value of the tilt angle signal of the hip joint
hip_tilt_angle_last	Last value of the tilt angle signal of the hip joint
knee_tilt_angle_mean	Mean value of the tilt angle signal of the knee joint
knee_tilt_angle_std	Standard deviation of the tilt angle signal of the knee joint
knee_tilt_angle_max	Maximum value of the tilt angle signal of the knee joint
knee_tilt_angle_min	Minimum value of the tilt angle signal of the knee joint
knee_tilt_angle_range	Range of the tilt angle signal of the knee joint
knee_tilt_angle_first	First value of the tilt angle signal of the knee joint
knee_tilt_angle_last	Last value of the tilt angle signal of the knee joint
ankle_yaw_angle_mean	Mean value of the yaw angle signal of the ankle joint
ankle_yaw_angle_std	Standard deviation of the yaw angle signal of the ankle joint
ankle_yaw_angle_max	Maximum value of the yaw angle signal of the ankle joint
ankle_yaw_angle_min	Minimum value of the yaw angle signal of the ankle joint
ankle_yaw_angle_range	Range of the yaw angle signal of the ankle joint
ankle_yaw_angle_first	First value of the yaw angle signal of the ankle joint
ankle_yaw_angle_last	Last value of the yaw angle signal of the ankle joint
ankle_tilt_angle_mean	Mean value of the tilt angle signal of the ankle joint
ankle_tilt_angle_std	Standard deviation of the tilt angle signal of the ankle joint

ankle_tilt_angle_max	Maximum value of the tilt angle signal of the ankle joint
ankle_tilt_angle_min	Minimum value of the tilt angle signal of the ankle joint
ankle_tilt_angle_range	Range of the tilt angle signal of the ankle joint
ankle_tilt_angle_first	First value of the tilt angle signal of the ankle joint
ankle_tilt_angle_last	Last value of the tilt angle signal of the ankle joint

Human Subjects:

thigh_angle_mean	Mean value of the tilt angle signal of the thigh
thigh_angle_std	Standard deviation of the tilt angle signal of the thigh
thigh_angle_range	Range of the tilt angle signal of the thigh
thigh_angle_first	First value of the tilt angle signal of the thigh
thigh_angle_last	Last value of the tilt angle signal of the thigh
thigh_ang_vel_mean	Mean value of the tilt angle signal of the thigh
thigh_ang_vel_std	Standard deviation of the tilt angular velocity signal of the thigh
thigh_ang_vel_range	Range of the tilt angular velocity signal of the thigh
thigh_ang_vel_first	First value of the tilt angular velocity signal of the thigh
thigh_ang_vel_last	Last value of the tilt angular velocity signal of the thigh
shank_angle_mean	Mean value of the tilt angle signal of the shank
shank_angle_std	Standard deviation of the tilt angle signal of the shank
shank_angle_range	Range of the tilt angle signal of the shank
shank_angle_first	First value of the tilt angle signal of the shank
shank_angle_last	Last value of the tilt angle signal of the shank
shank_ang_vel_mean	Mean value of the tilt angular velocity signal of the thigh
shank_ang_vel_std	Standard deviation of the tilt angular velocity signal of the shank
shank_ang_vel_range	Range of the tilt angular velocity signal of the shank
shank_ang_vel_first	First value of the tilt angular velocity signal of the shank
shank_ang_vel_last	Last value of the tilt angular velocity signal of the shank
foot_angle_mean	Mean value of the tilt angle signal of the foot
foot_angle_std	Standard deviation of the tilt angle signal of the foot
foot_angle_range	Range of the tilt angle signal of the foot
foot_angle_first	First value of the tilt angle signal of the foot
foot_angle_last	Last value of the tilt angle signal of the foot
foot_ang_vel_mean	Mean value of the tilt angular velocity signal of the foot
foot_ang_vel_std	Standard deviation of the tilt angular velocity signal of the foot
foot_ang_vel_range	Range of the tilt angular velocity signal of the foot
foot_ang_vel_first	First value of the tilt angular velocity signal of the foot
foot_ang_vel_last	Last value of the tilt angular velocity signal of the foot
torso_tilt_angle_mean	Mean value of the tilt angle signal of the torso
torso_tilt_angle_std	Standard deviation of the tilt angle signal of the torso
torso_tilt_angle_range	Range of the tilt angle signal of the torso
torso_tilt_angle_first	First value of the tilt angle signal of the torso
torso_tilt_angle_last	Last value of the tilt angle signal of the torso
torso_tilt_ang_vel_mean	Mean value of the tilt angular velocity signal of the torso
torso_tilt_ang_vel_std	Standard deviation of the tilt angular velocity signal of the torso

torso_tilt_ang_vel_range	Range of the tilt angular velocity signal of the torso
torso_tilt_ang_vel_first	First value of the tilt angular velocity signal of the torso
torso_tilt_ang_vel_last	Last value of the tilt angular velocity signal of the torso
torso_rotation_angle_mean	Mean value of the rotation angle signal of the torso
torso_rotation_angle_std	Standard deviation of the rotation angle signal of the torso
torso_rotation_angle_range	Range of the rotation angle signal of the torso
torso_rotation_angle_first	First value of the rotation angle signal of the torso
torso_rotation_angle_last	Last value of the rotation angle signal of the torso
torso_rotation_ang_vel_mean	Mean value of the rotation angular velocity signal of the torso
torso_rotation_ang_vel_std	Standard deviation of the rotation angular velocity signal of the torso
torso_rotation_ang_vel_range	Range of the rotation angular velocity signal of the torso
torso_rotation_ang_vel_first	First value of the rotation angular velocity signal of the torso
torso_rotation_ang_vel_last	Last value of the rotation angular velocity signal of the torso

The following tables report the selected features for the final models. Each model was assigned a different color: green for ‘direction_ft’, red for ‘sts_trs_ft’, blue for ’transition_type’ and orange for ‘steady_data_type’.

Prediction Human

event_thigh_angle_mean	event_foot_angle_mean	opposite_shank_angle_mean	torso_tilt_angle_mean
event_thigh_angle_std	event_foot_angle_std	opposite_shank_angle_std	torso_tilt_angle_std
event_thigh_angle_range	event_foot_angle_range	opposite_shank_angle_range	torso_tilt_angle_range
event_thigh_angle_first	event_foot_angle_first	opposite_shank_angle_first	torso_tilt_angle_first
event_thigh_angle_last	event_foot_angle_last	opposite_shank_angle_last	torso_tilt_angle_last
event_thigh_ang_vel_mean	event_foot_ang_vel_mean	opposite_shank_ang_vel_mean	torso_tilt_ang_vel_mean
event_thigh_ang_vel_std	event_foot_ang_vel_std	opposite_shank_ang_vel_std	torso_tilt_ang_vel_std
event_thigh_ang_vel_range	event_foot_ang_vel_range	opposite_shank_ang_vel_range	torso_tilt_ang_vel_range
event_thigh_ang_vel_first	event_foot_ang_vel_first	opposite_shank_ang_vel_first	torso_tilt_ang_vel_first
event_thigh_ang_vel_last	event_foot_ang_vel_last	opposite_shank_ang_vel_last	torso_tilt_ang_vel_last
event_shank_angle_mean	opposite_thigh_angle_mean	opposite_foot_angle_mean	torso_rotation_angle_mean
event_shank_angle_std	opposite_thigh_angle_std	opposite_foot_angle_std	torso_rotation_angle_std
event_shank_angle_range	opposite_thigh_angle_range	opposite_foot_angle_range	torso_rotation_angle_range
event_shank_angle_first	opposite_thigh_angle_first	opposite_foot_angle_first	torso_rotation_angle_first
event_shank_angle_last	opposite_thigh_angle_last	opposite_foot_angle_last	torso_rotation_angle_last
event_shank_ang_vel_mean	opposite_thigh_ang_vel_mean	opposite_foot_ang_vel_mean	torso_rotation_ang_vel_mean
event_shank_ang_vel_std	opposite_thigh_ang_vel_std	opposite_foot_ang_vel_std	torso_rotation_ang_vel_std
event_shank_ang_vel_range	opposite_thigh_ang_vel_range	opposite_foot_ang_vel_range	torso_rotation_ang_vel_range
event_shank_ang_vel_first	opposite_thigh_ang_vel_first	opposite_foot_ang_vel_first	torso_rotation_ang_vel_first
event_shank_ang_vel_last	opposite_thigh_ang_vel_last	opposite_foot_ang_vel_last	torso_rotation_ang_vel_last

Prediction Robot

event_hip_rot_angle_mean	event_knee_tilt_angle_mean	opposite_hip_rot_angle_mean	opposite_knee_tilt_angle_mean
event_hip_rot_angle_std	event_knee_tilt_angle_std	opposite_hip_rot_angle_std	opposite_knee_tilt_angle_std
event_hip_rot_angle_max	event_knee_tilt_angle_max	opposite_hip_rot_angle_max	opposite_knee_tilt_angle_max
event_hip_rot_angle_min	event_knee_tilt_angle_min	opposite_hip_rot_angle_min	opposite_knee_tilt_angle_min
event_hip_rot_angle_range	event_knee_tilt_angle_range	opposite_hip_rot_angle_range	opposite_knee_tilt_angle_range
event_hip_rot_angle_first	event_knee_tilt_angle_first	opposite_hip_rot_angle_first	opposite_knee_tilt_angle_first
event_hip_rot_angle_last	event_knee_tilt_angle_last	opposite_hip_rot_angle_last	opposite_knee_tilt_angle_last
event_hip_yaw_angle_mean	event_ankle_yaw_angle_mean	opposite_hip_yaw_angle_mean	opposite_ankle_yaw_angle_mean
event_hip_yaw_angle_std	event_ankle_yaw_angle_std	opposite_hip_yaw_angle_std	opposite_ankle_yaw_angle_std
event_hip_yaw_angle_max	event_ankle_yaw_angle_max	opposite_hip_yaw_angle_max	opposite_ankle_yaw_angle_max
event_hip_yaw_angle_min	event_ankle_yaw_angle_min	opposite_hip_yaw_angle_min	opposite_ankle_yaw_angle_min
event_hip_yaw_angle_range	event_ankle_yaw_angle_range	opposite_hip_yaw_angle_range	opposite_ankle_yaw_angle_range
event_hip_yaw_angle_first	event_ankle_yaw_angle_first	opposite_hip_yaw_angle_first	opposite_ankle_yaw_angle_first
event_hip_yaw_angle_last	event_ankle_yaw_angle_last	opposite_hip_yaw_angle_last	opposite_ankle_yaw_angle_last
event_hip_tilt_angle_mean	event_ankle_tilt_angle_mean	opposite_hip_tilt_angle_mean	opposite_ankle_tilt_angle_mean
event_hip_tilt_angle_std	event_ankle_tilt_angle_std	opposite_hip_tilt_angle_std	opposite_ankle_tilt_angle_std
event_hip_tilt_angle_max	event_ankle_tilt_angle_max	opposite_hip_tilt_angle_max	opposite_ankle_tilt_angle_max
event_hip_tilt_angle_min	event_ankle_tilt_angle_min	opposite_hip_tilt_angle_min	opposite_ankle_tilt_angle_min
event_hip_tilt_angle_range	event_ankle_tilt_angle_range	opposite_hip_tilt_angle_range	opposite_ankle_tilt_angle_range
event_hip_tilt_angle_first	event_ankle_tilt_angle_first	opposite_hip_tilt_angle_first	opposite_ankle_tilt_angle_first
event_hip_tilt_angle_last	event_ankle_tilt_angle_last	opposite_hip_tilt_angle_last	opposite_ankle_tilt_angle_last

Recognition Human

Recognition Robot