"Locomotion intention detection for lower-limb amputees by using Dynamic Bayesian Networks"

Draguet, Camille

ABSTRACT

The recent development of powered lower-limb prostheses allows patients suffering from a lower-limb amputation to participate in daily life activities. However, the use of such a prosthesis requires the user to be able to control it in real time, in order to adapt the device to his locomotion modes. Multiple strategies exist but pattern recognition, in particular, has shown promising results for real-time control. To this end, the subsequent work investigates the implementation of a Dynamic Bayesian Network that is built based on data from a small number of mechanical sensors, worn by two able-bodied subjects. This method aims at distinguishing multiple locomotion modes such as the level-ground walking mode, the standing mode, the stair ascent and descent modes, as well as the ramp ascent and descent modes. The experiments that are exposed in this thesis were carried out on a limited number of subjects. Given the small amount of data collected, the classification results obtained were insufficient. However, promising outcomes are presented in the Discussion chapter and the Dynamic Bayesian Network managed to classify tasks in a very short delay of approximately 0.00145 [s]. Therefore, this Master Thesis paves the way for future studies about locomotion intention detection through the use of a Dynamic Bayesian Network.

CITE THIS VERSION

UCLouvain

École polytechnique de Louvain

Locomotion intention detection for lower-limb amputees by using Dynamic Bayesian Networks

Author : **Camille Draguet**
Supervisor : **Renaud Ronsse**
Readers : **Michel Verleysen, Ali H. Al-Dabbagh**
Academic year 2019–2020
Master [120] in Biomedical Engineering

Abstract

The recent development of powered lower-limb prostheses allows patients suffering from a lower-limb amputation to participate in daily life activities. However, the use of such a prosthesis requires the user to be able to control it in real time, in order to adapt the device to his locomotion modes. Multiple strategies exist but pattern recognition, in particular, has shown promising results for real-time control. To this end, the subsequent work investigates the implementation of a Dynamic Bayesian Network that is built based on data from a small number of mechanical sensors, worn by two able-bodied subjects. This method aims at distinguishing multiple locomotion modes such as the level-ground walking mode, the standing mode, the stair ascent and descent modes, as well as the ramp ascent and descent modes. The experiments that are exposed in this thesis were carried out on a limited number of subjects. Given the small amount of data collected, the classification results obtained were insufficient. However, promising outcomes are presented in the Discussion chapter and the Dynamic Bayesian Network managed to classify tasks in a very short delay of approximately 0.00145 [s]. Therefore, this Master Thesis paves the way for future studies about locomotion intention detection through the use of a Dynamic Bayesian Network.

Acknowledgements

I would like to thank all the people who have contributed in one way or another to this Master Thesis.

First, I would like to thank my promotor, Mr. Renaud RONSSE, for his supervision, advice and availability throughout the year.

Mr. LEE for the reading and suggestions regarding the section about the Dynamic Bayesian Networks.

Ms. Virginie OTLET and Mr. François HEREMANS for their help relative to the use of the sensors.

Ms. Dounia MULDERS and Mr. Cyril DE BODT for the informative discussion about Probabilistic Graphical Models and Dynamic Bayesian Networks.

Mr. Victor COUPLET for his help in the implementation of Dynamic Bayesian Networks.

I would also like to thank Ms. Sophie COUPLET for agreeing to be my subject in experiments involving long hours of walking, ramps and stairs climbing. A special thanks also to Mr. Edouard COUPLET for his involvement, the supervision of the experiments when I was the subject and the numerous discussions linked to Machine Learning concepts.

Finally, a warmly thank to Ms. Teresa YATES, Mr. Philippe DRAGUET and Ms. Daphné GODFIRNON for their excellent work of proofreading.

Contents

Glossary

ANN Artificial neural network.

AO Adaptive oscillator.

BCI Brain-computer interface.

BN Bayesian network.

CNN Convolutional neural network.

CNS Central nervous system.

CPG Central pattern generator.

DAG Directed acyclic graph.

DBN Dynamic bayesian network.

DR Dimensionality reduction.

EEG Electroencephalography.

EMG Electromyography.

fNRIS Functional near infrared spectroscopy.

GMM Gaussian mixture model.

HC Heel contact.

IMU Inertial measurement unit.

KNN K-nearest neighbors.

LDA Linear discriminant analysis.

LW Level-ground walking.

MAP Maximum a posteriori.

MI Mutual information.

MST Mid-stance.

MSW Mid-swing.

PCA Principal component analysis.

PGM Probabilistic graphical model.

PNS Peripheral nervous system.

QDA Quadratic discriminant analysis.

RD Ramp descent.

SA Stair ascent.

SD Stair descent.

ST Standing.

SVM Support vector machine.

TMR Targeted muscle re-innervation.

TO Toe-off.

vGFR Vertical ground force reaction.

List of Figures

List of Tables

Introduction

Lower-limb amputations can be a traumatic event in the life of a person. In addition to seeing their morphology completely changed, patients that undergo lower-limb amputations have to deal with strong consequences on their daily life activities. They suffer from a constant struggle with complex and demanding environments composed of stairs, ramps, and so forth. Lower-limb amputees also face loss of social integrity due to their difficulty in participating in all kinds of activities. In some countries, amputees even suffer from diminished social dignity [1].

There are many conditions that can cause an amputation: trauma, cancer, birth malformation, vascular diseases, and so on. In particular, diabetes is recognized as one of the most important causes of lower-limb amputations. The risk of amputation depends also on other factors such as ethinicity and access to health care [2].

Over the past few years, researchers have identified the number of lower-limb amputations in the world. When taking into account the two types of lower-limb amputations, transfemoral (above the knee) and transtibial (under the knee) amputations, the numbers range from 5.8 to 31 over a total population of 100 000 people. This means that in some countries 31 persons per 100 000 have undergone a lower-limb amputation. These numbers considerably rise when only considering people suffering from diabetes (46.1 to 9600 over 100 000). These statistics were extracted from a study carried out in 2011 [2]. Today however, the numbers are even higher, mainly due to the development of diabetes. The number of lower-limb amputations, whatever the kind, is expected to nearly double by the year 2030 [3].

Based on these numbers, it can be estimated that a significant part of the population can benefit from a lower-limb prosthesis to help them to carry out their day-to-day activities. The recent development of active lower-limb prostheses enables lower-limb amputees to deambulate at a lower metabolic cost compared to passive prostheses. While passive prostheses are typically composed of springs that can only store and release energy, active lower-limb prostheses have the ability to generate net power thanks to the use of actuators to model knee and ankle joints [4]. One of the major challenges when developing an active prosthesis is to enable it to be controlled by the user in real-time. This implies the design of a control interface, allowing the user to communicate with the prosthesis. It also implies that the prosthesis should adapt its behavior to the user's intents.

Multiple strategies have been considered for the control of the prosthesis by the user. Generally available active prostheses imply the use of a key fob. Transitions with such prostheses require the patient to slow down, stop and press a button in order to transition from one activity mode to the other. Some strategies also imply performing a particular and exaggerated body movement so that the prosthesis sensors are able to detect it [5]. In

echo control approaches, the prosthetic leg mimics the kinematic of the unimpaired limb, requiring a complex instrumentation of both legs [5]. On the contrary, intent recognition algorithms have the advantage of providing seamless, automatic and natural transitions [6], generally without having to instrument the sound leg. This last approach will be investigated in this Master Thesis.

This work aims first at investigating the methods for detecting the locomotion intentions of a user that have been proposed in the literature. An algorithm will then be implemented. This algorithm has been selected as the one yielding the best possible performance in terms of classification accuracy. An other criterion was that this algorithm had to be applicable for real-time use. The final objective is to determine whether, on the basis of signals from a small number of portable sensors, this algorithm is capable of differentiating among multiple locomotion tasks. The tasks under investigation are level-ground walking, ramp ascent and descent, stairs ascent and descent, as well as the standing mode. Those tasks have been selected because they are part of many activities of everyday life. Besides, these particular tasks are studied in many of the articles reviewed. The experiments will be conducted on healthy subjects in order to implement a general algorithm. Subsequently, this algorithm will be used to control an active leg prosthesis.

This Master Thesis is structured as follows. Chapter 1 will address a theoretical background related to the human locomotion. After briefly presenting the neural path of the locomotion intents, the elements of the gait cycle and its characteristics for a healthy subject will be detailed.

Chapter 2 will present a review of the literature that is intended to be as comprehensive as possible. This chapter starts by summarizing several articles on the topic of locomotion intention detection. It then focuses on the selection and comparison of the studies that have been reviewed.

In Chapter 3, all the materials and methods are described. This chapter explains in detail the experiments that have been conducted. It exposes the dimensionality reduction and features selection methods that have been investigated. It also gives a theoretical background on the algorithm that has been selected for implementation in Chapter 2.

Finally Chapter 4 exposes the classification results and the possibilities for real-time application, while Chapter 5 discusses these results.

Chapter 1

The human gait cycle

This chapter aims at giving a brief theoretical background on the human gait cycle. It first exposes the neural pathways that are taken by the motor commands in order to induce a locomotion task. Then a decomposition of the human gait cycle and the gait events that punctuate this cycle are exposed.

1.1 Origin of locomotion

The human control of locomotion is a complex research domain that still needs a lot of investigation to be fully understood. However, some general structures are recognized as being implicated in the locomotion control such as the central pattern generator and the reflex arc. However, to understand what those elements are, one needs to take a step back and take a look at how the human nervous system works.

In the human nervous system, there are two major parts: the central nervous system (CNS), composed of the brain and the spinal cord, and the peripheral nervous system (PNS), composed of the sensitive nerves. The CNS is responsible for the analysis and integration of sensory and motor information while the PNS is mainly responsible for connecting the CNS to the limbs and organs [7].

The PNS can be further decomposed into the sensory compounds, including the connections between the sensory receptors and the CNS, and the motor compounds. The motor part of the PNS, which is of interest in the locomotion control, can also be broken down into the somato-motor system and the autonomous system. The somato-motor system is composed of the motor nerves connecting the brain and the spinal cord to the skeletal muscles while the autonomous system is composed of nervous cells that control the functions of the viscera, such as the heart or lungs.

Two pathways connect the CNS and the PNS. The afferent pathway is the path taken by the nerve impulse to get to the CNS from the PNS, when a sensory feedback is sent back to the CNS for example. The efferent pathway is the path taken by the nerve impulse to get from the CNS to the PNS, when an order is sent from the CNS [7]. While using the efferent pathway, the movements can be produced by different sources. The central pattern generator (CPG) is responsible for basic motor pattern. Volitional movements originate at the cortical level and reflexes are controlled by the reflex arc neural pathway where neurons synapse in the spinal cord. Human locomotion, in particular, depends on basic patterns generated by the CPG but is also fine-tuned at the cortical level (e.g. initiation of gait) and by the reflex-arc (efficiency and stabilisation) [5]. The control loop

for human locomotion is observable in Figure 1.1.

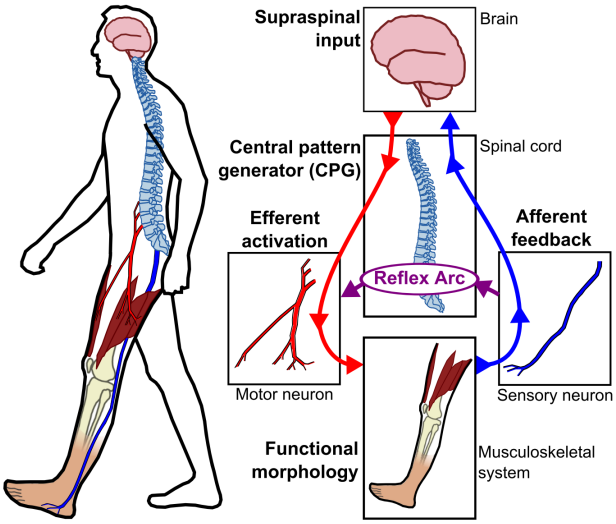


Figure 1.1: Sensory-motor control loop for human locomotion. From [5].

As explained, the human locomotion originates at various points in the nervous system. The CPG is one of them and is responsible for basic patterns such as rhythmic motions. Indeed, one of the main characteristic of the human locomotion is its periodicity. This property is of major importance for this particular work. The motions that are observable during human locomotion are repeated over and over. The general set of movements that are repeated cyclically during human walking is called the gait cycle. A complete decomposition of this gait cycle will be done in the next section.

1.2 Analysis of the gait phases

The explanations that are given in this section stand for a normal gait, meaning that the subject does not suffer from any pathology or malformations that could alter the way he or she walks.

First of all, one has to make a clear distinction between the terms step and stride. A stride refers to a complete gait cycle while a step refers only to the part between the moment when a foot hits the ground and the moment when the other foot hits the ground. These concepts are illustrated in Figure 1.2.

Typically, the stride length ranges between 1.33 to 1.63 [m] in individuals with a normal gait [8]. The step length ranges between 0.61 to 0.9 [m] and is generally longer for men than for women [8]. The cadence of walking is often defined as the number of steps per minute but some authors prefer to define it as the number of strides per minute [9]. By keeping this second definition, the walking speed can be directly linked to the cadence with the following Equation 1.1 [9].

$$\text{walking speed} = \text{cadence} \cdot \text{stride length} \tag{1.1}$$

Since the average walking speed is estimated to range from 1.4 to 1.49 [m/s] [8], the average cadence ranges approximately from 52 to 67 strides per minute.

The self-selected walking speed is the speed that an individual will automatically adopt while performing daily activities involving walking. This self-selected walking speed is

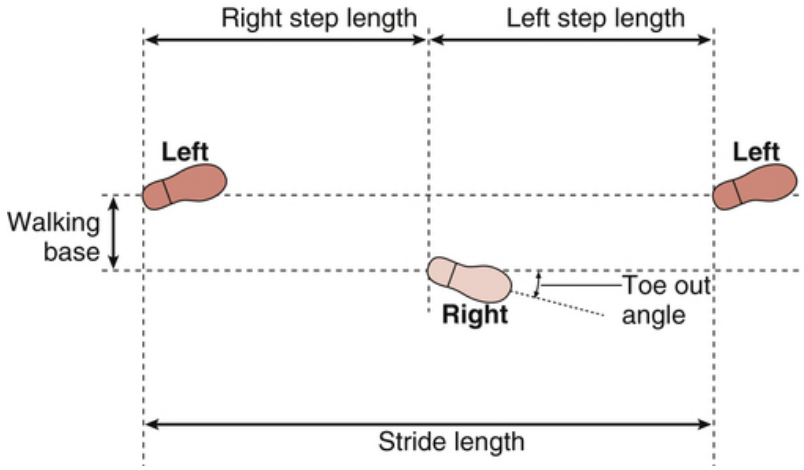


Figure 1.2: Illustration of the difference between stride and step. From [8].

generally slower for women than for men and that comes with a slightly longer stance phase. This concept is explained in the next paragraph. It is important to keep in mind that measurements about gait are widely dependent on the walking speed and that ways of walking may also be somewhat different among individuals. Therefore a generalization had to be made for this Master Thesis work. For the sake of convenience, the different cadences proposed for the experiments were defined in terms of steps per second. Indeed, since the rhythm was set by a metronome, it was easier for the subjects to take a step at the beat of this metronome.

The general gait cycle starts at the first contact of the reference foot (for the sake of this research, the right foot was used as reference) on the ground and ends at the next occurrence of that event. Usually, the gait cycle is divided in two phases: the stance phase, which is the part of the cycle where the reference foot is in contact with the ground, and the swing phase, which refers to the part where the reference foot is off the ground. The stance phase last for approximately 60% of the gait cycle and the swing phase for 40%. This imbalance can be explained by the relatively short period of time that a human spends in double support when walking.

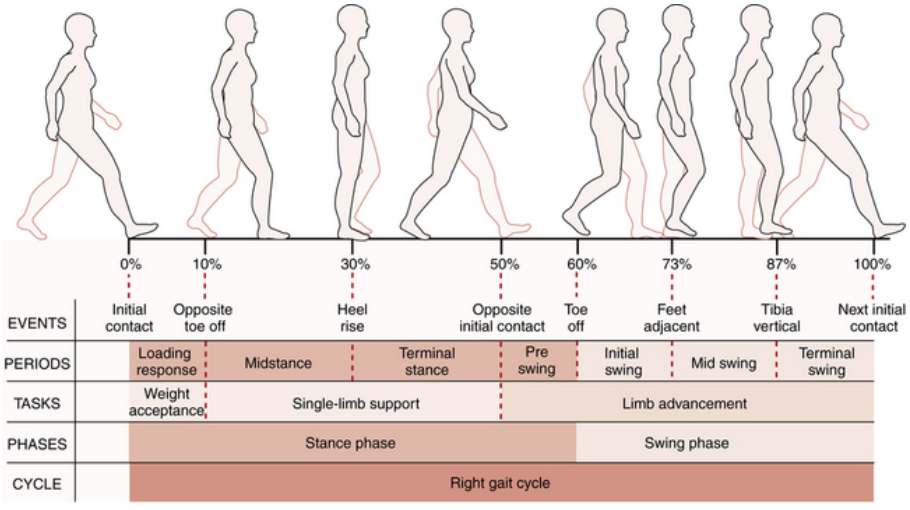


Figure 1.3: Phases and events occurring during a gait cycle. From [8].

As observable in Figure 1.3, two tasks main are accomplished during the stance phase: weight acceptance consists in the transfer of body mass from one limb to the other and single-limb support consists in supporting the whole body mass with the limb that is in stance phase, while the other limb is moving forward. During the swing phase, however, only one task is fulfilled and consists in moving the leg forward until reaching the point of contact with the ground.

Multiple events occur during the course of a gait cycle. For this particular work, 4 of them will be used as decision points for the algorithm. They are subsequently defined (from [10]):

- The heel strike or heel contact (HC) corresponds to the moment when the foot touches the ground, as the name suggests, at the end of the previous swing phase. This first contact is established with the heel. The heel contact is defined here as the initiation of the gait cycle.
- The mid-stance (MST) corresponds to the point where the body weight passes vertically over the supporting limb. This event is generally experienced at 30% of the gait cycle. In the first half of the stance phase, the non-supporting limb advances over the stationary foot until the body weight is aligned over the forefoot. This particular point will be designated as the mid-stance event.
- The toe-off (TO) corresponds to the moment when the toe leaves the ground. It is generally experienced at 60% of the gait cycle and marks the end of the stance phase.
- The mid-swing (MSW) corresponds to the mid-stance of the other leg (which is now the supporting limb). It is defined as the moment when the swing leg passes next to the stance leg and is generally experienced between 0.75% and 0.85% of the gait cycle. For this work purpose, a fixed moment had to be selected and thus the mid-swing was precisely defined at 75% of the gait cycle.

These events are highlighted in the following figure.

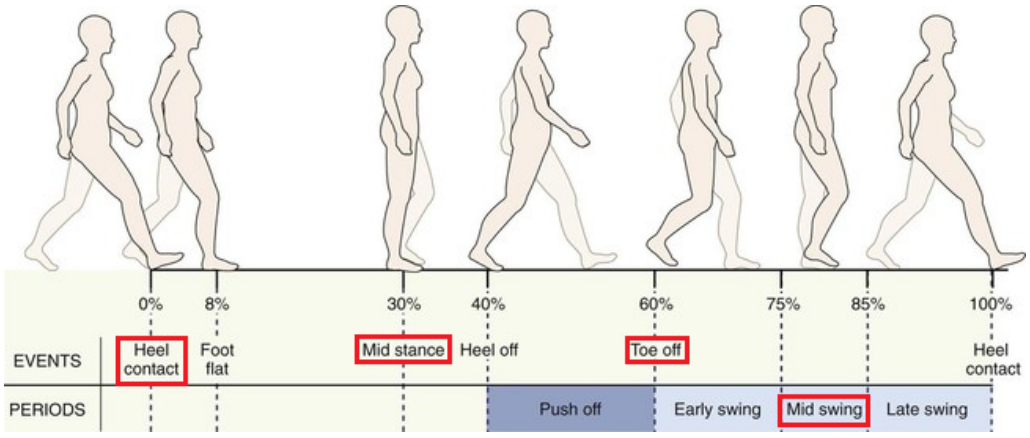


Figure 1.4: Events during the course of a gait cycle. Those that will be used as decision points in this work are framed in red. Adapted from [10].

Chapter 2

Walking intention detection: a review of the existing techniques

This chapter aims first at presenting the state of the art about locomotion intention detection. A comparison between the articles presented will then be carried out and a particular study will be selected based on its performances. The method proposed in this particular study will be developed and discussed in this Master Thesis.

In the framework of this Master Thesis, we investigated different strategies to help an active lower-limb prosthesis to estimate the walking intentions of a user. This includes multiple locomotion tasks such as level-ground walking, stair ascent/descent, ramp ascent/descent, etc. Multiple studies have been conducted on this topic and the first step of this work was, therefore, to review them and to compare the strategies employed. The articles investigated were published between 2008 and 2019. Most were drawn from the review addressed by Tucker et al [5] in 2015. More recent articles were found by browsing other bibliographies or by searching on dedicated platforms such as Google Scholar, Scopus, and so forth. On these platforms, more recent researches (2016-2019) about walking intention detection by lower-limb amputees were found.

2.1 Signals

To estimate the user intention among multiple locomotion tasks, mainly two kinds of signals can be recorded : neural signals and mechanical signals. In the following subsections, these signals will be explained as well as the methods to record them. One could also mention the possible use of environment sensing to detect the walking intention of the user. However, this is usually made in combination with the sensing of neural and/or mechanical signals and is out of the scope of this Master Thesis.

2.1.1 Neural signals

As locomotion originates at different levels, neural activity can be recorded at different levels to allow the detection of user's intention. Here are some strategies reviewed by Tucker et al. [5].

Brain-computer interfaces (BCI) record the neural electrical activity at the cortical level, thus in the central nervous system. Their major advantage is their potential to allow for a wide variety of volitional movements. There are different types of BCI such as functional near-infrared spectroscopy (fNRIS), which uses optical light to sense the haemodynamic response of the brain, electroencephalography (EEG), which consists in an array of elec-

trodes placed on the scalp to record the brain electrical activity, implanted electrode arrays, etc. All these methods have particular drawbacks. The implanted electrode array is highly invasive and the fNRIS and EEG are subject to motion artifacts. Moreover, there are two additional major drawbacks that are common to all BCI methods. First, the brain is responsible for a wide variety of movements and it may be complicated to distinguish between different cortical activities. Secondly, the human locomotion is still mainly controlled by the reflexive path in the spinal cord and by recording only the cortical activity, some important information may be missed [5].

If the electrical activity is recorded closer to the muscle, in the peripheral nervous system, the motor command becomes more specific and information about all control sources can be recorded. Moreover, there is a small delay between the nerve impulse and the generation of force in the muscle. This small delay can be used to design a controller ahead of the walking action. These signals can be collected through the use of electromyography (EMG). In surface EMG, electrodes are placed on the skin at the location of the muscle of interest, usually through palpation. Surface EMG is subject to changes in electrode-skin conductivity, motion artifacts, fatigue of muscles and cross-talk between nearby muscles. Therefore it requires a calibration each time the device is used [5].

2.1.2 Mechanical signals

By using mechanomyography (MMG), one could estimate the force production in a muscle by sensing the sounds or vibrations at the surface of skin. This could be done by placing accelerometers or microphones on the user's skin over the location of the muscle belly. This method has been shown to be less sensitive to muscle fatigue [11] than EMG but is very sensitive to motion artifacts [5].

Inverse dynamics can also be used to compute torques of joints from measurements of joints positions, as well as external forces applied on the limbs. This method uses the Newton-Euler equations to compute torques. The positions of joints and limb-segment orientations can be measured with numerous kinds of sensors including goniometers, inclinometers, accelerometers, gyroscopes, magnetometers and inertial measurement units (IMUs). Ground reaction forces can be measured among others by using instrumented insole. Moreover, when performing experiments on amputees, interaction forces between the user's body and the prosthesis can be measured by using load cells, strain gauges, pressure sensors and force-sensitive resistors [5].

2.2 Main classification algorithms

Not only is it possible to use multiple types of signals to detect the user walking intention but it is also possible to use different algorithms. The algorithms that will be presented thereafter, were drawn from the literature and are dedicated to the specific problem investigated in this Master Thesis. These algorithms are only briefly described but the selected method will be further detailed in Chapter 3.

Linear Discriminant Analysis (LDA) is a dimensionality reduction technique that can be extended for classification purposes. It is similar to Principal Component Analysis (PCA, that will be detailed in Chapter 3) and is often compared to PCA in terms of performances. LDA transforms the feature space and projects data to a lower dimensions subspace where the components of this subspace maximize the separation between classes [12]. Quadratic Discriminant Analysis (QDA) is a quadratic version of the LDA and relaxes some of its requirements [13].

While using a K-nearest neighbors (KNN) algorithm, a similarity measure is used to find

the k training examples that are the most similar to a target example that needs to be classified. A majority vote is performed among the k-nearest neighbors and the class that wins this majority vote is attributed to the target example. Numerous distance metrics can be used to assess similarity between examples like the Euclidean distance function or the Hamming distance function.

Support Vector Machines, or SVMs, are essentially binary classifiers but they can be generalized to multi-class classification by training multiple classifiers. SVMs use decision surfaces (hyperplane) defined in a feature space to perform classification. Data is implicitly mapped to the feature space, which is not the original space. The selected hyperplane is the one that maximizes the margin (the distance) between each class. In order to implement n-class SVMs, one can either train n classifiers, such that each classifier is trained for one particular class against all other classes, or train n(n-1)/2 classifiers, such that each classifier discriminates between a pair of classes. It is also possible to extend the SVM formulation to multiple categories but is maybe a more complex task [14].

Dynamic Bayesian Networks (DBNs) are an extension of Bayesian Networks that allow to integrate time history information. These are a type of Probabilistic Graphical Models (PGMs) that use graphical structures for inferring conditional probabilities about variables represented by nodes in the graphical structure. Bayesian Networks and DBNs are represented by directed acyclic graphs in which each edge between two nodes symbolizes a relation between the two corresponding variables. At the core of Bayesian Networks and Dynamic Bayesian Networks are probability laws. More particularly, Bayes' law and conditional independence property are heavily involved in inferences from these networks.

Artificial Neural Networks (ANNs) are more or less complex structures composed of different layers. Low-level features are fed in the input layer. The network performs combinations, convolutions and max-pooling operations of these low-level features to create high-level features in what is called the hidden layers. At the output layer, probabilities are available, which describe the expectations for the input sample of belonging to any of the possible classes.

2.3 Review of the methods presented in the literature

These algorithms and signal types were discussed in the reviewed papers. This section will briefly describe the main elements from each article. For the sake of clarity and comprehensiveness, these paragraphs will be organized as follows: studies will be grouped based on the implemented algorithms and in the case where multiple implementation were tested, the one leading to the best results will be chosen as reference. After presenting all papers relative to one particular algorithm, a table will present a summary of the main elements.

Before presenting all the reviewed papers, some concepts need to be clarified. First of all, it is important to understand that after recording the signals from EMG sensors or mechanical sensors, features of these signals need to be extracted. Such features characterize a given sample and are the attributes or the variables of this sample. Samples or observations are time analysis windows that are taken from the signals at particular moments. Usually, researchers use a sliding window of 100 to 400 [ms] that moves along the signal and stops to take a snapshot of it. These stops are marked according to the fixed increment of the sliding window. These concepts are illustrated in Figure 2.1.

Often researchers are also only interested in snapshots taken at particular gait events. In this case, the timing window is taken just before the gait event or spans this gait event

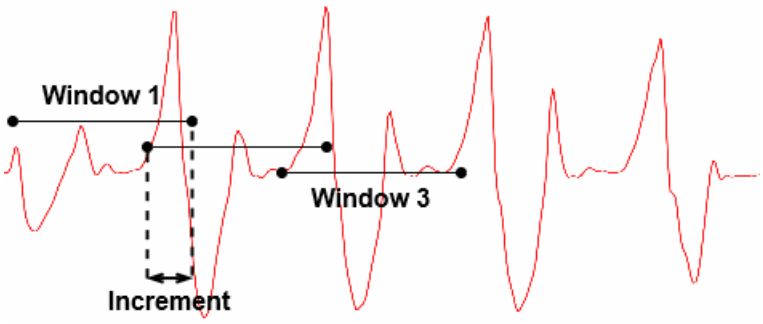


Figure 2.1: Sliding windows and increment between those sliding windows over a general signal.

and continue for 90 [ms] (for example) after the gait event. This last scenario induces a delay of 90 [ms], meaning that the classifier decision is taken approximately 90 [ms] after the gait event.

Besides, in these papers, the authors refer to classifier performances in terms of the steady-state classification error and the transitional classification error. The first represents the percentage of steps not occurring at a transition between locomotion modes, and that were misclassified, while the second is the percentage of transition steps that were misclassified [15]. Some articles also present the performance in terms of overall accuracy. This is a combination of the two aforementioned measures that is computed differently depending on the experimental circuit. A possible formulation would be

$$\text{Overall accuracy} = 1 - \frac{n_{\text{ss steps}} \cdot \text{ss error} + n_{\text{trans steps}} \cdot \text{trans error}}{n_{\text{steps}}} \tag{2.1}$$

In this equation, $n_{\text{ss steps}}$ is the number of steady-state steps that occurred during the experiment, $n_{\text{trans steps}}$ is the number of transitional steps that occurred during the experiment and n_{steps} is the total number of steps.

It is also worth mentioning that some of the reviewed papers present their methods for dimensionality reduction and feature selection. Some also discuss an optimal selection of window lengths and increments. However, the following paragraphs mainly focus on the choice of the type of signals and the choice of algorithms. Additionally, performance of these algorithms are presented as well as the delay necessary for the classifier decision, when mentioned.

2.3.1 Studies implementing a LDA

In their article *A strategy for identifying locomotion modes using surface electromyography* [16], Huang et al. investigated surface electromyography and pattern recognition combination for user's intents recognition. The tasks included level-ground walking, stepping over an obstacle, ascending and descending stairs, ipsilateral and controlateral turning and standing still. The experiments included both able-bodied subjects and transfemoral amputees. In this study, the authors stated that one of the main challenges of using EMG for intent recognition is that signals from EMG show large variations for a same task mode and therefore overlap between features could interfere with intents classification. Moreover, using neural contents may not be suited for patients suffering from a high level of amputation. Such patients could however benefit from targeted muscle re-innervation (TMR). To overcome the first issue, small timing windows were considered for pattern

recognition such that only small changes could be observed in signal contents over the course of the window. From these timing windows, features were extracted and passed into an LDA and a fully-connected two layers ANN. Each classifier was built for 4-phase windows defined pre and post heel contact and toe-off. ANN always outperformed LDA, even if the differences in classification scores were not statistically significant. LDA was however preferred because it was faster to train and less complex. The overall accuracy errors in the 4 phases were $12.4\% \pm 5.0\%$, $6.0\% \pm 4.7\%$, $7.5\% \pm 5.1\%$ and $5.2\% \pm 3.7\%$ for the 8 able-bodied subjects.

A. J. Young et al. [17] investigated the training of a user-independent classifier. Indeed, in general, the presented algorithms require a substantial amount of training data and recording that data for each new patient is a heavy burden. The authors used data from a pool of lower limb amputees performing walking, as well as ramp and stair ascent and descent to train classifiers and extend these classifiers to new patients. To do so, they recorded EMG and mechanical data. However, they obtained poor generalization results (accuracy of 48%) while using EMG data in combination with mechanical data against 62% while using mechanical sensors only. This raised the question of similarity of EMG patterns between different amputees. However, by introducing very little data from the new patient, including EMG data, the overall accuracy was improved to 86%. In this article, a LDA classifier was used.

The authors from [3] evaluated a classifier performance based on mechanical data alone, EMG data alone and a neuromechanical sensor fusion. They designed experiments were transtibial amputees subjects had to perform 6 different tasks including level-ground walking, ramp ascent, ramp descent, stair ascent, stair descent and a standing mode used as a baseline position. In their experiments, they investigated 10 common mode transitions. An LDA classifier was used to discriminate among tasks. The authors discovered that the neuromechanical sensor fusion yielded a transitional classification error of $2.3 \pm 0.7\%$ against $7.8 \pm 0.9\%$ for the mechanical sensors alone and $20.2 \pm 2.0\%$ for the EMG sensors alone. The classification decisions were always taken at toe-off and the features necessary to build the model and to perform inference were taken from a timing window spanning this gait event. Since the timing window was only extended to 125 [ms] after the gait event, the authors contended that there remained sufficient time to adapt the prosthesis parameters before the beginning of the next stance phase.

In [18] the authors trained their classifier based on mechanical sensors only. Training data collected from two single timing windows at heel contact and at toe-off provided better classification results than building models on training data collected from multiple timing windows sliding along the training signals. Indeed, this last approach may introduce unwanted errors since most signals present non-stationary behaviors during a gait cycle. In their article, the authors discussed multiple parameters such as the window size, the number of windows where each was used as a sample for the classifier and the number of repetitions of the training circuit. They implemented two classifiers, an LDA trained on data recorded from analysis windows before heel contact and an LDA trained on data recorded from analysis windows before toe-off. The best classification results were observed for a single window of 250 [ms] taken before each selected gait event. Besides, transitional and steady states results were significantly improved by performing each training circuit up to 5 times. By including the transition steps in the training data, the LDA reached an overall accuracy of 93.9%.

Varol et al. [19] studied a control architecture and a real-time intent recognition model

that was able to discriminate among standing, sitting and walking tasks. The sensors recording joint angles, angular velocities, interactions forces and torques were placed directly on the prosthesis. Those last two measures were taken between the users and the prosthesis and between the prosthesis and the environment. In this research, the authors used Gaussian Mixture Models (GMM), which employ the concepts of sub-populations and probability distributions in order to infer properties of the sub-populations given information about the population distribution. GMMs have the strong advantage of being able to approximate both simple (LDA, QDA) and complex models by setting the number of mixtures used. The number of mixtures as well as the dimensionality reduction method that were exploited, were used as search parameters and a total of 30 models were therefore compared. Finally, a 3-components LDA (for dimensionality reduction) and a GMM with 6 mixtures were selected. This method identified all activity-mode transitions intended by the user without any noticeable delay. However, the algorithm also identified 3 activity-mode that were not intended by the user. Nevertheless, the subject stated that he did not perceive these errors.

Table 2.1: Summary of the studies implementing LDA classifiers

Authors	Year	Sensors type	Performance	Delay (ms)	Comments
H. Huang et al. [16]	2009	EMG	Overall error $7.775 \pm 4.625\%$	x	
A.J. Young et al. [17]	2013	EMG + Mechanical sensors	Overall accuracy 86%	x	User independent study
D.C. Tkach et al. [3]	2013	EMG + Mechanical sensors	Transitional error $2.3 \pm 0.7\%$	125	
A.J. Young et al. [18]	2014	Mechanical sensors	Overall accuracy 93.9%	x	
H.A. Varol et al. [19]	2010	Mechanical sensors	x	x	Use of GMM algorithm, depending on the number of mixtures, it can approximate simpler or complex models

2.3.2 Studies implementing a KNN

In their article [20], Khademi et al. studied the performance of an enhanced KNN algorithm to discriminate among 4 locomotion modes: standing, slow walking, normal walking and fast walking. They used mechanical sensors to collect data and obtained a classification accuracy of 96%. The best classification accuracy was obtained while the algorithm was able to produce its decision every 0.0298 [s]. This delay needs to be reduced for real-time application.

Table 2.2: Summary of the studies implementing KNN classifiers

Authors	Year	Sensors type	Performance	Delay (ms)	Comments
Khademi et al. [20]	2015	Mechanical sensors	Overall accuracy 96%	29.8	

2.3.3 Studies implementing a SVM

In [21] a locomotion-mode identification based on neuromuscular-mechanical fusion was investigated. EMG signals were combined with ground reaction forces and moments measured at the junction of the leg and the prosthesis by lower-limb amputees. The authors emphasized the fact that a real-time intention detection is of prior importance to ensure patient safety. As mechanical sensors respond to patient movements while EMG signals precede the movement, EMG signals are often firstly investigated. However, in the previous study, the authors reached approximately 90% accuracy, which is promising but needs

to be improved to ensure patient safety. Therefore, they considered using a data fusion (EMG + mechanical signals) for user intentions detection but the delay introduced by this fusion needs to be evaluated. The tasks under investigation were level-ground walking, stepping over an obstacle, stair ascent and descent, as well as ramp ascent and descent. Two multiclass *one-against-one* SVMs were implemented for classification, one in each gait phase. For a k-class classification problem, $k(k-1)/2$ binary classifiers are built. 15 binary SVM were thus built for this 6-class classification problem. The class mode with the most votes out of the 15 decisions was chosen. An enhanced voting strategy was used to prevent unusual transitions from occurring by increasing the number of voting points when an unusual transition was detected. The authors demonstrated that neuromechanical fusion always outperformed the use of EMG signals or of mechanical signals alone. Classification accuracies of 99% and 95% were obtained, respectively in the stance phase and in the swing phase. Only transitions from level-ground walking and towards level-ground walking were considered and the system always detected these transitions before the critical timing occurred, the critical timing being defined as the last contact of the prosthesis foot with the ground and the first contact of the prosthesis foot with the ground, respectively.

Table 2.3: Summary of the studies implementing SVM classifiers

Authors	**Year**	**Sensors type**	**Performance**	**Delay (ms)**	**Comments**
H. Huang et al. [21]	2011	EMG + Mechanical sensors	Overall accuracy 99% (stance phase) 95% (swing phase)	x	

2.3.4 Studies implementing a DBN

In their article *An intent recognition strategy for transfemoral amputee ambulation across different locomotion modes* [15], Young et al. investigated the use of a two-timeslice DBN to classify level-ground walking, ramp and stair ascent and descent tasks. Since two time steps are considered for inference, the current task can therefore be computed from current observations and priors from the previous step. The features were computed from mechanical and EMG sensors. Transitional errors were relatively high (11.3%) when the classifier was trained on all five locomotion modes. However, by classifying ramp tasks as level-ground walking, the authors managed to obtain a transitional error of 0.9 % and a steady state error of 0.3%. This may be explained by the fact that ramp steps present similar patterns to level-ground walking steps. However, the authors stated that, as the ramp descent mode is substantially different in terms of prosthesis parameters, this classifier should be used in combination with a second method allowing to discriminate at least between level-ground walking and ramp descent modes. They suggested using a slope estimator or a second stage classifier, which would only be active while the prosthesis is in level-ground walking mode.

A.J. Young also participated in a study where the performances of a DBN and of an LDA were compared [22]. These comparisons were made for models built on mechanical data alone, EMG data alone and a combination of both sensor types. Mechanical sensors included IMUs, position and velocity sensors as well as load cells. Like for several previously presented articles, it was the combination of mechanical and EMG data that yielded the best results. Data was collected on 8 transfemoral amputees performing circuits including walking, ramps and stairs. The use of DBN with history information reduced the steady state classification errors from 3.0% (for the LDA) to 1.0% and the transitional classification errors from 18.4% to 12.2%.

Spanias et al. [23] presented an adaptive online method for user's intent recognition based on the recording of EMG data. The algorithm was continuously learning to incorporate new neural information as the subjects were deambulating. The experiments were conducted on 8 transfemoral amputees over several days, which highlighted the changing behavior of neural signals. Despite these changes, the 8 mode-specific DBNs implemented managed to classify 98% of the transitional steps and 99.5% of the steady-state steps. These results were obtained while using an analysis window overlapping the gait events by 90 [ms]. Mode-specific classifiers are classifiers that are trained for a particular mode. For example, a classifier could be trained to be active in level-ground walking and to be able to detect transitions from level-ground walking to another locomotion mode. The decisions are taken at specific gait events.

The classifier architecture used in [24] is the same as in the previously presented article [23]. However, in this study, the authors investigated intent recognition based on mechanical sensors alone that included IMUs, load cells, etc. They evaluated the performance of the DBNs on analysis windows of 300 [ms] and presented two approaches: a non-delayed system where the analysis window was taken before gait events and a delayed system where the analysis window overlapped the gait event by 90[ms]. The overall error observed for the non-delayed system was 0.3% and 0.98% for the delayed system. In order to obtain such results, the authors decided to classify ramp ascent tasks as level-ground walking. They estimated that these two locomotion modes were sufficiently similar to not cause any noticeable disturbance while not discriminated.

In [6], the authors used a DBN to explore the added value of incorporating time history information for intention recognition. In this study, 6 transfemoral amputees performed circuits including walking, as well as ramp and stairs tasks. They compared the integration of time history information by using a majority vote strategy, a DBN and a baseline where no time history information was added. These algorithms were fed with feature data from 13 mechanical sensors including IMUs, load cells, potentiometers and encoders, located on the prosthesis of each subject. For the DBN strategies, 8 models, one for each selected point during a gait cycle, were implemented and fed with features from 300 [ms] windows preceding these points. A minimum of 19% for the transitional error and of 1.5% for the steady state error was obtained for the DBN strategy while selecting specific points (among the 8 possibilities) in the gait cycle.

Young et al. also investigated a user-independent strategy that alleviates the burden of recording a substantial amount of training data for each new subject [25]. Additionally, they investigated a partially-dependent strategy where only little training data from the new subject was recorded. These two strategies were tested with a DBN and compared with a baseline strategy including 2 LDA, one at heel contact and one at toe-off. Subjects, who were all transfemoral amputees, performed multiple trials on circuits including walking, ramps and stairs. Data were recorded from mechanical sensors and signals were segmented into 300 [ms] analysis windows preceding 8 points during the gait cycle, as in the article previously presented. Mode-specific classification, where the previously detected locomotion task determined the next classifier to be used, was also tested. The combination of mode-specific and time history classification yielded the best results with a transitional error of 15% and a steady-state error of 12% for the user-independent strategy. Besides, this strategy reached a transitional classification error of 9% and a steady-state classification error of 3% for the user-dependent trials.

Across-day experiments were studied by Simon et al. in [26]. Signals recorded during

sessions a few days apart may be subject to large variations and it is therefore necessary to build a reliable classifier that integrates these possible variations. Features were extracted from signals coming from a pool of embedded mechanical sensors. The authors used the same classifier architecture as in [24] and those classifiers were used to discriminate among 5 ambulation modes. Transitions between modes could only occur 90 [ms] after one of the gait events, namely heel contact, mid-stance, toe-off and mid-swing. 4 sessions were conducted, each of them including different circuits and tasks. The recording of data from these 4 sessions was spread over a maximum of 4.5 months. The overall error decreased from $1.45 \pm 0.3\%$ when the system was trained with data from session 1 and tested with data from session 4, to $0.6 \pm 0.2\%$ when the system was trained with data from sessions 1, 2 and 3 and tested on session 4.

EMG data has the potential to improve user control of a prosthetic leg. It is indeed valuable data for pattern recognition algorithms and has been proved to allow seamless transitions between locomotion modes. However, the availability of such signals strongly depends on the level of amputation and on opportunities for re-innervation of the amputated limb. In this context, Hargrove et al. [27] have studied the impact of nerve transfers on the performance of an algorithm that allowed to discriminate between multiple knee and ankle movements. Nerve transfer is analogous to targeted muscle re-innervation surgery and consists in sewing pieces of the remaining nerve to a muscle motor point. After several months of training, discrete contractions in the injured muscle could be observed. This kind of surgery could make it possible to build a classifier based on neuromechanical fusion in a wider range of leg amputation conditions. As a reminder, neuromechanical sensor fusion has been shown to outperform both EMG sensors alone and mechanical sensors alone. It is also interesting to point out that the authors of this study used a DBN for classification. This DBN was trained with data from re-innervated muscles and mechanical sensors. They obtained a classification accuracy of 92% while including knee flexion, knee extension, ankle plantarflexion and ankle dorsiflexion. Tibial rotation and femoral rotation were also considered.

2.3.5 Studies implementing an ANN

In [28], the authors investigated the use of a powered-ankle foot prosthesis to assist transtibial amputees in daily activities including level-ground walking and stairs descent. Myoelectric signals recorded from the gastrocnemius and the tibialis anterior, which are muscles strongly implicated in locomotion patterns, are used to transition the prosthesis from level-ground walking to stairs descent and vice versa. Two separate finite state controllers were implemented, one for each particular task. Finite state controllers are widely used to control leg prostheses since gait patterns are repetitive between strides. Mechanical sensors including potentiometers and force transducers were placed on the prosthesis to record ankle angle, torque and foot contact pressure and other signals that are used to detect states and to transition from one state controller to the other. Switching between the two finite state controllers requires detecting the user's intent. This was done through the use of a Myoelectric Processing Unit, by detecting particular muscle activity patterns. Standard deviations of the myoelectric signals taken on 100 [ms] sliding windows were fed into a feed-forward neural network composed of a single hidden layer. This neural network demonstrated its ability to detect the correct locomotion intent. However, this article was mainly focused on the design of finite state controllers and did not give any classification results.

The study from Gupta et al. [1] explored the use of multiple classifiers trained on time

Table 2.4: Summary of the studies implementing DBN classifiers

Authors	Year	Sensors type	Performance	Delay (ms)	Comments
A.J. Young et al. [15]	2013	EMG + Mechanical sensors	Steady-state error 0.3% Transitional error 0.9%	x	Results obtained while classifying both ramp ascent and descent as level-ground walking
A.J. Young et al. [22]	2014	EMG + Mechanical sensors	Steady-state error 3.0% Transitional error 12.2%	x	
Spanias et al. [23]	2018	EMG	Steady-state accuracy 99.5% Transitional accuracy 98%	90	
A.M. Simon et al. [24]	2017	Mechanical sensors	Overall error 0.98% (non-delayed) 0.3% (delayed)	90	Two experiments: non-delayed and delayed decisions (90 ms)
A.J. Young et al. [6]	2014	Mechanical sensors	Steady-state error 1.5% Transitional error 19%	x	
A.J. Young et al. [25]	2016	Mechanical sensors	Steady-state error 3% (user dependent) 12% (user indenpendent) Transitional error 9% (user dependent) 15% (user independent)	x	User-independent strategy investigation. However, results for the user-dependent trials were given.
A.M. Simon et al. [26]	2018	Mechanical sensors	Overall accuracy $0.6 \pm 0.2\%$	x	Across-day study
L.J. Hargrove et al. [27]	2013	EMG	Overall accuracy 92%	x	With muscles re-innervation

domain and frequency domain features from EMG. Their study shows that time domain features are best suited for intent recognition. They also presented a greedy feature selection method and the algorithms yielded better results while working with fewer features. This study included discrimination between walking, stair ascent and stair descent. The algorithms investigated were a SVM, an LDA, an ANN, a DT (decision tree), a KNN and an NB (Naïve Bayes). By using the greedy feature selection, only 9 features were selected for the neural network and this feature vector reached a classification accuracy of $94.30 \pm 1.64\%$, which was the best score obtained in this study.

The following article, *Human activity recognition from accelerometer data using convolutional neural network* [29], is worth mentioning due to its high classification results obtained through the use of a Convolutional Neural Network (CNN), which is a type of ANN. The authors used their CNN to discriminate among walking, running and staying still. A particularity of this study is the use of data from accelerometers from user smartphones. The CNN was a 1D network composed of a convolutional layer, a max-pooling layer, a fully-connected layer and a softmax output layer. The best classification results were obtained while feeding the 20th feature into the 1D CNN, which yielded an accuracy of 92.71%. Although this study does not fall exactly within the scope of this work, these interesting results obtained through a CNN may also be achieved while adapting the method to intent recognition by lower limb amputees.

Another study that highlights the use of CNN for locomotion intent recognition is the study conducted by Y. Feng et al. in 2019 [30]. The authors used the signals from a strain gauge to discriminate among 3 and 5 locomotion modes: ascent mode, descent mode and level-ground walking for the 3-mode discrimination and ramp ascent, ramp descent, stair ascent, stair descent and level-ground walking for the 5-mode discrimination. The CNN was composed of multiple convolutional and max-pooling layers. The CNN's performances were evaluated by using a 5-fold cross-validation and this approach reached $88.86 \pm 4.23\%$

for the 5-mode discrimination. However, this method was such that the decision was taken at the next step (approximately 0.522 [s][2] delay for a classical gait cadence).

Table 2.5: Summary of the studies implementing ANN classifiers

Authors	Year	Sensors type	Performance	Delay (ms)	Comments
S. Au et al. [28]	2008	EMG	x	x	
R. Gupta et al. [1]	2018	EMG	Overall accuracy $94.30 \pm 1.64\%$	x	
L. Song-Mi et al. [29]	2017	Mechanical sensors	Overall accuracy 92.71%	x	CNN Smartphones data
Y. Feng et al. [30]	2019	Mechanical sensors	Overall accuracy $88.86 \pm 4.23\%$	522.5	CNN Only sensor used was a strain gauge

2.3.6 General summary

From the previous paragraphs, it can be shown that LDA and DBN are widely used algorithms for locomotion mode recognition and often yield good classification results. However, CNN, SVM and, to a lesser extent, KNN have also proved efficient for particular experiments and applications. Therefore, all these algorithms should be considered as potential solutions for the following steps of this work.

Among the above-mentioned researches, the use of EMG data, mechanical data and a neuromechanical fusion is fairly evenly split. However, most of the articles come to the conclusion that neuromechanical fusion yields the best classification results. The use of EMG data is much discussed. Indeed, EMG data brings important information for real-time classification, since neural signals precede the user's walking intentions. But EMGs also raise many concerns. They can vary due to electrode shifts, changes in electrode/skin impedance, loss of skin-electrode contacts and so forth. Classification errors could be induced by such variations. Nevertheless, these errors could be corrected by the use of an adaptive algorithm using mechanical signals in case of wide variations in EMG signals. This solution is proposed in [23]. On the other hand, mechanical data related to user's walking intentions only react to these intentions and may be less suited for real-time applications since they could induce small delays. However, given the numerous studies conducted with the use of mechanical data alone, one can estimate that this delay should not interfere with the users activities or may not even be noticed by these users. Besides, according to [3], the use of mechanical data alone could yield almost the same classification performances as using a neumomechanical fusion. This conclusion is also supported by [23], which demonstrated that despite the improvements brought by the use of EMG signals in combination with mechanical signals, those improvements were only marginal. Other approaches such as a well-chosen classifier, the use of thoughtful mechanical signals and so on, could bring similar benefits in a less cumbersome manner. Among the widely used mechanical sensors, inertial measurement units, load cells, potentiometers and encoders to measure the angular positions and angular velocities of the knee and ankle, etc. could be mentioned.

Of all these studies, some may be considered as possible ways to improve the work that will be done in this Master Thesis. These include notably [17] and [25] that aimed at investigating the performance of an algorithm on new users without subsequent heavy

[2]The authors mentioned a delay of one step. Indeed, the method requires a whole gait cycle to recognize the locomotion mode. Therefore, the delay was computed as step duration = step length/walking speed, where a normal step length ~ 0.755[m] and a normal walking speed ~ 1.445[m/s]

training. In [26], the authors implemented an algorithm that succeeded in classifying data from across-day sessions. This is of major importance for clinical applications of pattern recognition algorithms. The study conducted by Hargrove et al. [27] on a patient who benefited from targeted muscle re-innervation was also worth mentioning as a lead for generalizing methods based on neuromechanical fusion.

2.4 Comparing methods

One of the main problems in comparing papers is that the different authors do not use benchmarking to present and evaluate their experimental results. Therefore, in order to implement a rigorous comparison method, it was necessary to establish several comparison metrics, all calculated or extracted in the same way in each article. In order to do so, several assumptions had to be made. They will be explained in the following points. The two metrics that were chosen are the following:

- The computational time. This metric is an estimate of the computational cost generated by the analysis of the signals and their classification by the algorithm. The computational time will have a direct impact on the ability of the prosthesis to react in real time to the user's intentions. In most of the articles referenced above, no measure of the computational time was indicated. Therefore, a reasonable assumption would be that the classification is done at the end of the time analysis window, pending that the features calculated for each signal are computationally inexpensive. This hypothesis was put forward following an exchange with A.M. Simon, one of the main authors and co-author of many of the above-mentioned papers.
- The performance of the algorithm. For obvious reasons of safety and stability, the final objective is still to find a strategy that discriminates between the different user's walking intentions with as few errors as possible. In some of the articles listed above, the authors make a distinction between 2 kinds of possible errors: the steady-state classification errors and the transitional classification errors. Moreover, in some other articles, the overall accuracy is presented. In order to compare all the reviewed papers, one should distinguish the type of classification error presented and compare them according to this classification error type. A general formula for calculating the overall performance of the algorithm based on the transitional errors and the steady-state errors could also have been established. For this purpose, it would have been necessary to estimate the percentage of transitional steps and of steady-state steps in a random deambulation path including stairs ascents, level-ground walking, ramp descent, etc. The idea was however dropped because the percentage of transitional and steady-state steps mainly depends on the experiments and that experiments conducted in the various articles were different.

Initially, the goal of this Master Thesis is to re-implement, investigate and discuss some improvements regarding an experiment that has already proven itself. This experiment must be chosen as the one minimizing the discussed metrics. Among all the articles that were examined for the purpose of this research, only a few were selected for the comparison. There are many reasons not to use an article for the comparison. Indeed, either the information available was not sufficient to estimate the computational cost and performance of the algorithm, or the experience was conducted on a number of tasks that was not sufficient, or the research question diverged too much from the one under consideration, and so forth. However these articles are still a source of interesting information for the development of the classification algorithm and the validation of certain assumptions.

Moreover, some could also be taken into consideration for the elaboration of algorithm improvements and for a possible implementation of these improvements.

The following flowchart in Figure 2.2 shows the method and the criteria that were applied for selecting articles.

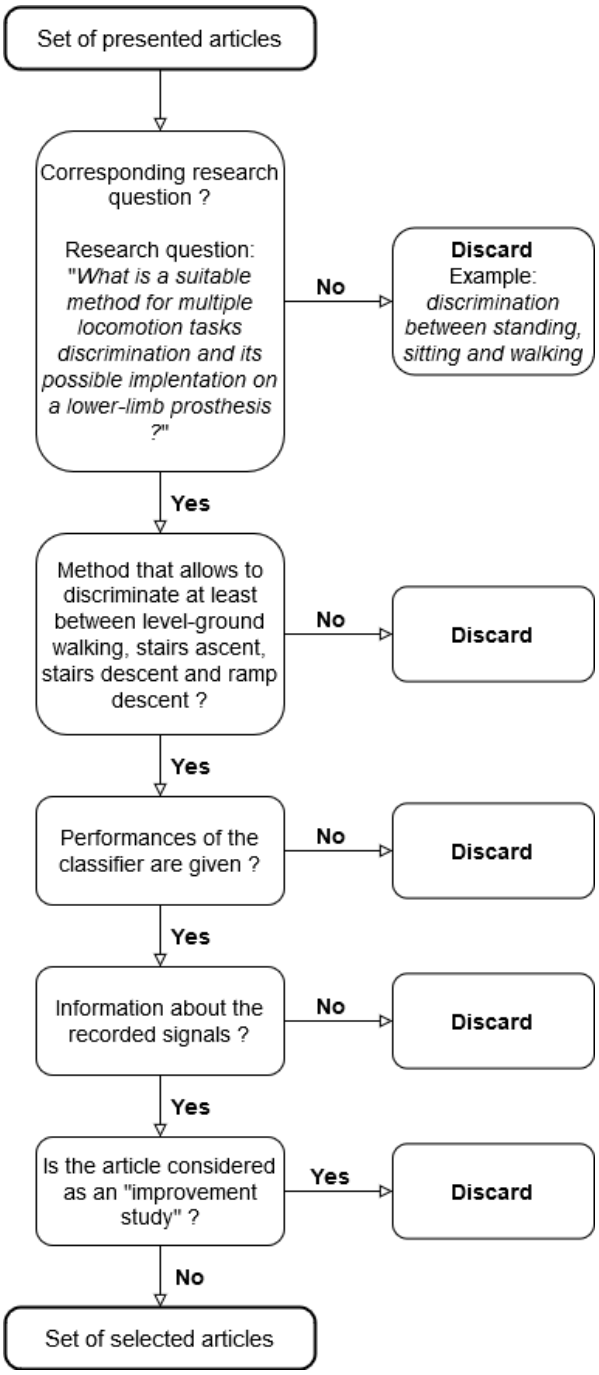


Figure 2.2: Flowchart for selecting articles. The studies considered as improvement studies are notably those implying user-independent strategies, across-day experiments and targeted muscle re-innervation.

This selection left us with a series of 7 articles, listed in Table 2.6. The computed metrics such as the time delay and the type of classification errors are also included in this table.

Table 2.6: Selected articles. The comparison metrics are presented for each article.

N°	Title	Main author	Year	Time delay (ms)	Overall error (%)	Steady-state error (%)	Transitional error (%)
1	Continuous locomotion-mode identification for prosthetic legs based on neuromuscular -mechanical fusion [1] [21]	H. Huang	2011	0	2.6	x	x
2	An intent recognition strategy for transfemoral amputee ambulation across different locomotion modes [15]	A.J. Young	2013	0	x	1.7	12.0
3	Intent recognition in a powered lower limb prosthesis using time history information [6]	A.J. Young	2014	0	x	1.5	19.0
4	A training method for locomotion mode prediction using powered lower limb prostheses [18]	A.J. Young	2014	0	6.1	5.0	17.0
5	Analysis of using EMG and mechanical sensors to enhance intent recognition in powered lower limb prostheses [22]	A.J. Young	2014	0	2.5	1.0	12.2
6	Delaying ambulation mode transition decisions improves accuracy of a flexible control system for powered knee-ankle prosthesis [24]	A.M. Simon	2017	0	0.98	0.78	2.65
				90	0.3	0.27	0.55
7	A strain gauge based locomotion mode recognition method using convolutional neural network [30]	Y. Feng	2019	522.5	11.14	x	x

[1] The performances were given in terms of accuracy in the swing phase and accuracy in the stance phase. The overall error has thus been computed as a weighted sum between the classification errors in the swing phase (40% of the gait cycle) and the classification errors in the stance phase (60% of the gait cycle) [31].

In order to find the experiment that minimizes the two objectives, the Pareto front was identified for the set of selected articles. This consists in finding the limit for which further minimizing one of the two objectives only results in the increase of the other objective. The Pareto front for the overall classification error is displayed in red in Figure 2.3. The Pareto fronts for the steady-state error and for the transitional error are available in Appendix A. As one can observe, both experimental methods explained in [24] are on the Pareto front and will thus be implemented during the course of this work. These methods will be further detailed in the following section. It is important to understand that for a more general comparison, other parameters such as low cost of sensors, ease of placement of those sensors, and so forth, should also be taken into account. However, for the purpose of this Master Thesis, we need to limit the comparative study.

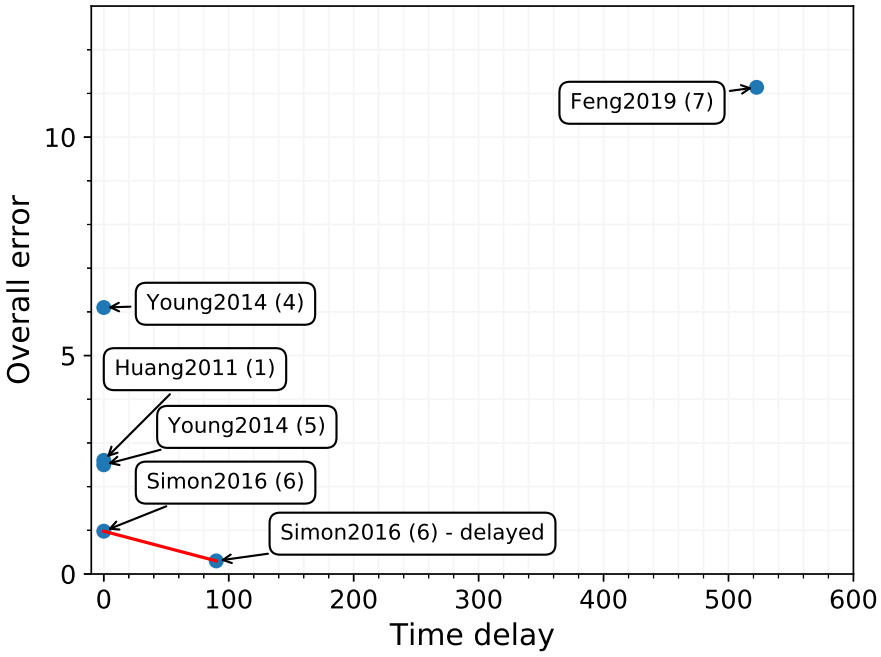


Figure 2.3: Set of selected articles, positioned according to their classification error and their time delay. The Pareto front is indicated in red.

2.5 Detailed presentation of the selected article

As mentioned above, thanks to the previous comparison, the methods from [24] will be implemented. The goal of this study was to observe the effects of delaying the ambulation mode recognition by a small amount of time. Therefore, they investigated the effects of a 90 [ms] delayed window on the accuracies of mode-specific classifiers. According to [11], this small delay was not supposed to affect the user's performances noticeably. Transitions between 6 modes were examined: level-ground walking, stair ascent, stair descent, ramp ascent, ramp descent and standing. The transitions between prosthesis modes were allowed at various discrete time points during the gait cycle, including heel contact, mid-stance, toe-off and mid-swing. Another concern raised in this study was the need for the classification algorithm to handle a large range of data to account for more variability in the signals. Indeed, in most studies the experiments are performed in a laboratory and the user is asked to perform specific tasks. This helps to create quite

an accurate system but it is complicated to take into account all the perturbations and the situations that could occur into a non-laboratory environment. The designed system needs to account for a large variability in speed, environment and obstacle approaches. Therefore, to assess the performances of the algorithm, the subject had the ability to freely deambulate in a laboratory environment composed of a 3-step staircase, a 4-step staircase, a level-ground walking surface and a 10 degree inclined surface.

The experimental protocol to record the training data consisted in a series of in-laboratory ambulation tasks such as:

- level-ground walking approaches towards and from ramp ascent or descent and level-ground walking approaches towards and from the two staircases ascent or descent
- climbing stairs in a stairwell
- climbing up two steps, standing, climbing-up two more steps, standing, turning around and descending the steps the same way
- walking at various speeds, straight-line walking and walking in circles
- standing, including shuffle steps, turning and quiet standing

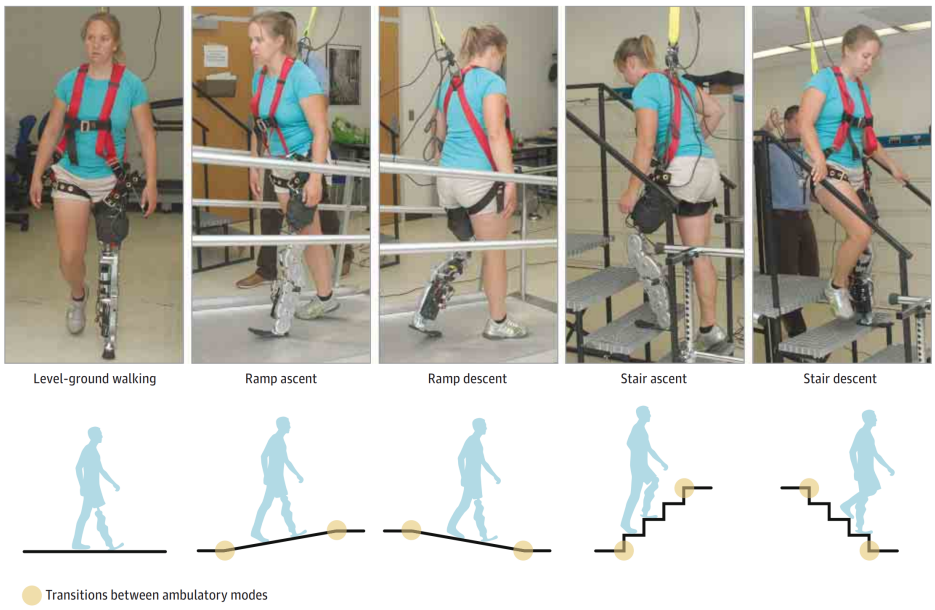


Figure 2.4: [Above] A transfemoral amputee wearing the powered knee-ankle prosthesis demonstrating various ambulation modes. [Below] Transitions investigated. From [32], which is an earlier research conducted by the authors and where subjects performed a similar experiment.

In order to account for more variability, the users were able to approach stairs or the ramp at different angles (0°, 45°or 90°). Besides, as these experiments were performed on amputees, the subjects led with their sound side for stair ascent approaches, the prosthesis side for stair descent and both for ramp ascent or descent.

For the experiments, the authors used a third generation powered knee-ankle prosthesis designed by the Vanderbilt University. This prosthesis weighs 4.2 [kg], which is comparable to the mass of a normal limb segment [24]. Knee and ankle joints are actuated by two motor-driven ball screws, translating the rotational motion of the motor to the required

linear motion with little friction [33]. The ankle actuation unit is also composed of a spring mounted in parallel with the ball screw. 120°-flexion can be reached at the knee and 45°-plantarflexion and 20°-dorsiflexion can be reached at the ankle. More information about the prosthesis is available in [4]. Those values correspond to the generally accepted ranges for a normal range of motion of knee and ankle [34, 35]. The data was recorded from mechanical sensors, all placed on the amputated leg, at a sampling frequency of 500 [Hz]. Most of these sensors were already integrated in the prosthesis. The mechanical signals, which are recorded, are the knee and ankle joint angles and velocities, the motor currents, the prosthesis acceleration and rotational velocity, the calculated thigh and shank inclination angles and the 6-degree of freedom forces and moments. Sensor data is evaluated in 30 [ms] increment windows. Two experiments were conducted. Data was segmented into 300 [ms] windows immediately preceding the gait events in the first experiment. This can be understood as follows: when the gait event is detected, thanks to the ground force reaction profile, obtained from the instrumented insole for example, only the sensor data from the last 300 [ms] is used by the classifier to estimate the walking intention. In the second experiment, the windows started 210 [ms] before the gait events and ended 90 [ms] after. In both case, the classification decisions were taken at the end of the timing window. For each timing window, the mean, the standard deviation, the maximum and the minimum values, as well as the initial and the final values were computed.

To control the prosthesis, joints torques are generated through an impedance-based approach. One can consider that the prosthesis has two possible states: a weight-bearing state and a nonweight-bearing state corresponding to stance and swing phase, respectively. In each state, knee and ankle are each modeled by a spring-damper system with a fixed equilibrium point. Therefore, the knee and ankle torques can be described by Equation 2.2.

$$\tau_i = k_i(\theta - \theta_{ei}) + b_i\dot{\theta} \tag{2.2}$$

where k_i, b_i and θ_{ei} are the linear stiffness, damping coefficient and equilibrium point, in the ith state. θ is the joint angle and $\dot{\theta}$ is the joint angular velocity.

The authors implemented 8 mode-specific classifiers. These classifiers are active in particular locomotion modes as shown in Table 2.7. In addition, classifiers were separated depending on the particular gait event at which they should take their decisions. In this implementation, level-ground walking and ramp ascent were treated as one class because the two modes have similar impedance parameter settings. Moreover, the toe-off classifier was not mode-specific because experimental data were missing for that particular case.

Table 2.8 indicates which classifier (e.g. HC_ST or TO_ST) will be responsible for the classification decisions according to the last gait event detected and to the previous locomotion mode in action. The colors indicated in the Table refer to the state machine in Figure 2.6. Red is for classifiers taking the decision at heel contact, green for classifiers taking the decision at mid-stance, blue for classifiers taking the decision at toe off and purple is for classifiers taking the decision at mid-swing.

Each mode-specific classifier was a DBN. As mentioned above, many of the experiments among those reviewed implemented that type of classifier. This particular network allows for the use of time history information for the classification. Indeed, the features extracted from mechanical signals change over the course of a gait cycle. It can thus be beneficial to capture these variations to improve the classification performance [6]. Moreover, some transitions are more likely to occur than others (e.g., level-ground walking to stair ascent vs stair descent to ramp ascent) which increases the improvements brought by the use

Table 2.7: Mode-specific classifiers. Table from (Simon, 2016 [24]). LW stands for level-ground walking mode, RD for ramp descent, SD for stair descent, ST for standing and SA for stair ascent.

Gait Phase	Classifier	Active in	Description	Number of Classes
Heel Contact	HC_LW	Level-ground walking	Predicted steady state and the transitions from level-ground walking to stair and ramp descent	3 (LW, RD, SD)
	HC_SD	Stair descent	Predicted steady state and the transitions from stair descent to level-ground walking	2 (LW, SD)
	HC_RD	Ramp descent	Predicted steady state and the transitions from ramp descent to level-ground walking	2 (LW, RD)
	HC_ST	Standing	Predicted steady state and the transition from standing to srair descent	2 (ST, SD)
Mid-Stance	MST_SD	Stair Descent	Predicted the transition from stair descent to level-ground walking when the first step on level ground was with the prosthesis (e.g., on 3-step staircase)	2 (LW, SD)
Toe Off	TO	Level-ground walking, ramp descent, stair descent	Predicted steady state and the transitions from level-ground walking to stair ascent and between level-ground walking and ramp/stair descent	4, (LW, RD, SA, SD)
	TO_ST	Standing	Predicted steady state and the transition from standing to stair ascent	2 (ST, SA)
Mid-Swing	MSW_SA	Stair Ascent	Predicted steady state and the transition from stair ascent to level-ground walking when the first step on level ground was with the sound side (e.g., on 3-step staircase)	2 (LW, SA)

Table 2.8: Classifiers presented according to the gait event and the active locomotion mode

	Standing	Level-Ground Walking	Stair Ascent	Stair Descent	Ramp Ascent	Ramp Descent
Heel Contact	HC_ST	HC_LW		HC_SD		HC_RD
Mid-Stance				MST_SD		
Toe Off	TO_ST	TO		TO		TO
Mid-Swing			MSW_SA			

of such a classifier. Bayesian techniques are powerful strategies that allow to take into account this time history information. One could use Hidden Markov Model but the use of such a model requires a stationary signal. A DBN, relaxing this stationary assumption, was thus implemented in [24]. More particularly, a two timeslices Bayesian network was implemented for classification, allowing for the use of current observations and prior probability distributions from the previous step. [6] gives a mathematical formula to compute the maximum a posteriori (MAP) estimate with the Bayes' law. The same strategy was used in [24]. The MAP corresponds to the class with the maximum posterior probability, which is computed as a combination of past and current information. These mathematical formulations and other concepts linked to the DBN are explained in Chapter 3. At all of the gait events indicated in Table 2.7, the DBN was able to calculate likelihood probabilities (current information) from the features extracted from the 300 [ms] time window. Prior probabilities were propagated from the previous gait event by multiplying the previous posterior probability with the probabilities of transitioning from the previous step to the current one. The class with the maximum posterior probability was chosen for the current time step.

In the results section of [24], it is stated that the users did not notice the 90 [ms] delay for some transitions at heel contact (e.g., between standing, level-ground walking, stair and ramp descent). However, users did notice a small transition delay at toe-off for transitions from standing to stair ascent, by experiencing a small delay in the update of

impedance parameters. This delay did not affect the use of the prosthesis however. The overall classification error was of 0.98 [%] in the non-delayed system and of 0.30 [%] in the delayed system (see Table 2.6). Thus, delaying classification decisions significantly reduced the overall error. Figure 2.5 indicates the difference in error rates between the delayed and non-delayed systems.

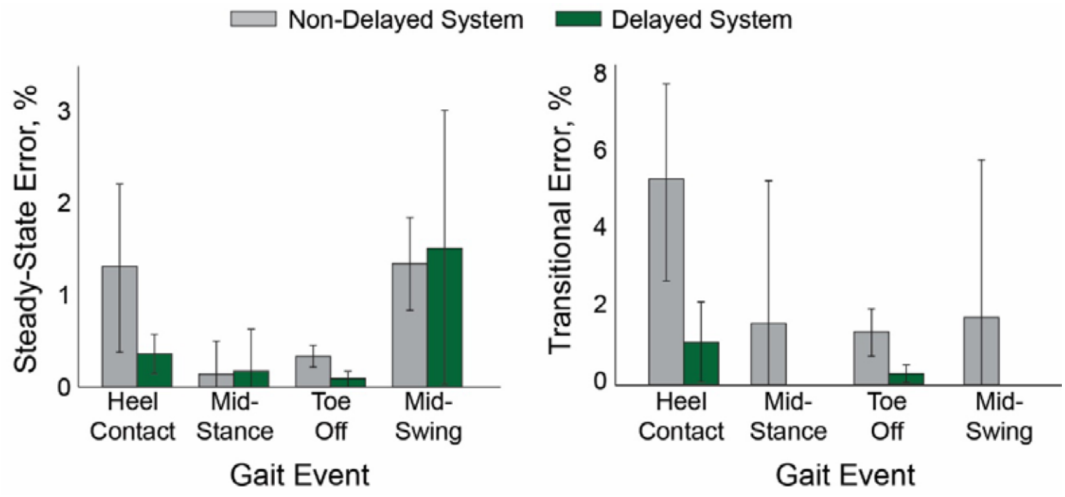


Figure 2.5: From [24]. Average effect of non-delayed vs delayed system on steady-state (left) and transitional (right) classification error. Heel contact error rates are the average of the HC_LW, HC_SD, HC_RD and HC_ST classifiers. Toe-off error rates are the average of the TO and TO_ST classifiers. Error bars indicate standard deviation. Note different vertical axis scaling between the left and right figures.

In the discussion, the authors explained that the users were able to perform a wider variety of transitions than those performed in the experimental protocol thanks to the enhanced state machine architecture. This architecture is presented in Figure 2.6. It shows that not all mode transitions were trained, and that those which were really unlikely to occur in real-time use were not allowed. When mechanical sensor thresholds and timers were sufficient to trigger the transitions between two modes (e.g. transition between standing and level-ground walking), these transitions were not included as a separate class in the pattern recognition system (no classifier discriminating between these two modes in Table 2.7).

Thanks to this system architecture, users were able to approach staircases or ramps at various speed and angles, with long or short steps, etc. However, it should be noted that the improved accuracies (compared to the previous strategy) were partly due to the addition of a 6-dof load cell mounted between the knee and the ankle on the prosthesis. The signal from a 6-dof load cell can not be recorded on healthy subjects on which the experiments of this Master Thesis will be conducted. Moreover, the computation of thigh and shank inclination angles also contributed to these results but were however not computed in this work. The classification of level-ground walking and ramp ascent in one unique class constitutes another improvement compared to previous studies such as [32, 22]. Indeed, in those studies, a differentiation between these two modes led to many misclassification errors.

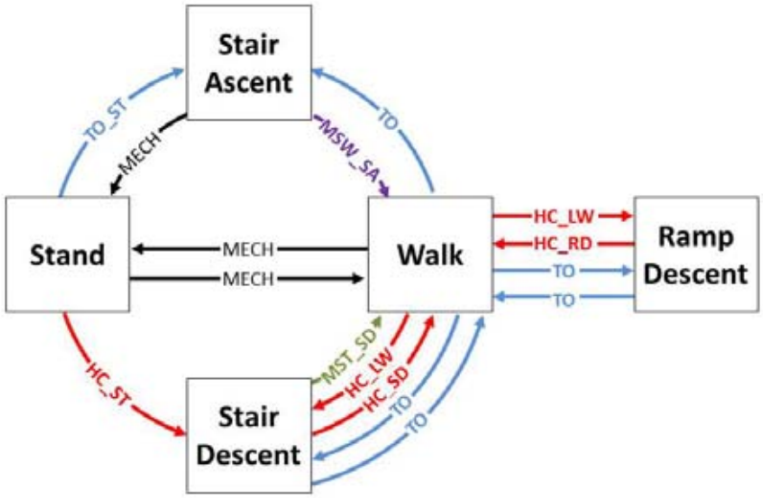


Figure 2.6: From [24]. An overview of the mode-specific classifier architecture. State machine prosthesis modes are shown indicating the corresponding classifier or mechanical trigger associated with each transition. Mechanical transitions are labeled in black, heel contact classifiers are in red, mid-stance classifier in green, toe-off classifiers are in blue, and mid-swing classifier in purple. Ramp ascent data was grouped together with walking data.

Finally, the choice of the 90 [ms] window could be discussed. While a larger delay could induce lower error rates, it may be perceived as more cumbersome by the user, impeding the use of the prosthesis. The system architecture presented here still needs to be tested in real-time with an online analysis.

2.6 General conclusion

Walking intention recognition is a problem that has been widely studied and many approaches are possible. Different types of signals and of algorithms could be used. Neuromechanical fusion has proven to be the most efficient for walking intention detection but EMG signals are not always available, depending on the level of amputation, and are subject to variations. Dynamic Bayesian Networks have been broadly implemented and tend to outperform the other strategies. In addition to these two main elements, numerous parameters may influence the classification results such as the choice of features, the size of the analysis window and so forth. In this work, a DBN built on mechanical features will be implemented and a thoughtful feature selection will be made. Nevertheless, subsequent work is needed to investigate the influence of the size of the timing window, the increment of the timing window, possible combination between mechanical and EMG features, etc.

It should also be mentioned that the study from Simon et al. [24] on which this work will be largely based has been conducted on transtibial amputees. However, the experiments conducted in the framework of this Master Thesis aim at developing a general algorithm, able to detect the locomotive intention of healthy subjects as a first step. Therefore, the sensors and protocol will be selected for that type of experiment. This excludes all types of sensors that necessarily need to be embedded on a prosthesis such as load cells. This intention detection module might be integrated later in the software environment of a prosthesis, allowing to control it.

Chapter 3

Materials and methods

In this chapter, all the materials that were used in the experiments and the methods that were implemented for the classification are presented. In particular, the theory relative to Dynamic Bayesian Networks is exposed at the end of this chapter.

3.1 Context

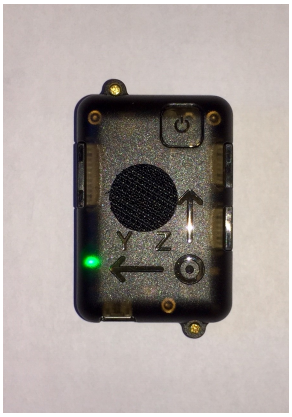
As a reminder, the objective of this Master Thesis is the implementation of an algorithm capable of differentiating among multiple locomotion tasks. As a first step, this algorithm will be tested on the deambulation of healthy subjects in an offline analysis. Further studies are necessary for a real-time implementation and for an extension to the control of a lower-limb prosthesis.

The following experiments were carried during the COVID-19 crisis. The experimental environment had to be modified multiple times and moved from an indoor space to an outdoor one. This particular situation also restricted the number of subjects participating in the experiments, as the experimenter could only work with people living under the same roof. Moreover, the use of one of the sensors required subjects to have a foot size of approximately 40. Only two subjects met that constraint. All these elements explain the small number of experiments and the author of this work is well-aware that this number is very small compared with what is found in the literature. It is also worth mentioning that the experimenter was herself the subject of her own protocol and that for this reason, an extra person had to be trained to use the sensors and to launch the data recording.

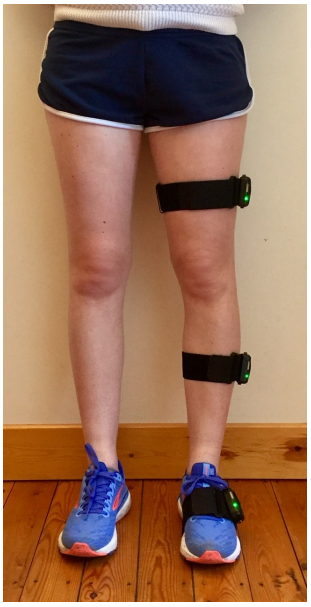
The investigated locomotion tasks can be encountered during daily activities and are therefore frequent. This is a good start to enable amputees to regain some mobility and participate in everyday social activities. The locomotion tasks that the classifier will be able to discriminate among are the still-standing position, level-ground walking, stair ascent and descent and ramp ascent and descent.

3.2 Sensors

In order to measure limb parameters during locomotion tasks, 3-IMUs (*NGIMU, x-IO Technologies*, Figure 3.1a) were placed on the left leg of each subject as depicted in Figures 3.1b and 3.1c. Each IMU was equipped with 3-dof sensors with sampling frequencies of 50 [Hz]. A 3-dof gyroscope recorded the angular position, a 3-dof accelerometer recorded the linear acceleration of the segment and finally, a 3-dof magnetometer recorded the magnetic field. The signals from this last sensor were not used for the task discrimination since



(a) NGIMU x-IO Technologies



(b) Front view

(c) Side view

Figure 3.1: IMUs placement

it was not useful for identifying gait patterns.

To record the vertical ground force reaction and the position of the center of pressure, a size 40 instrumented left-insole (*Feetme One, Feetme Inside*), equipped with 68 pressure sensors, was used to record signals at a sampling frequency of 100 [Hz].

Figure 3.2: Instrumented Insole, Feetme One

3.3 Experimental Protocol

Subjects had to perform circuits including stairs, a ramp and a level-ground walking surface at different gait cadences (the gait cadence being defined here as the number of steps per second): 0.8333 [Hz] (slow), 1.1667 [Hz] (normal) and 1.5 [Hz] (fast). Gait cadences were rythmed thanks to a metronome and randomized between experiments. Each experiment was thus conducted three times, once at each cadence. Prior to each data recording, subjects were allowed to perform the circuit as many times as they desired at the imposed gait cadence to get comfortable with the experimental setup.

Given the COVID-19 particular circumstances and restrictions linked to the size of the insole, only two subjects could participate in the experiments. Both subjects were healthy women whose characteristics are listed in Table 3.1.

Table 3.1: Characteristics of the subjects

	Age [y]	**Size [cm]**	**Weight [kg]**
Subject 1	22	169	64
Subject 2	56	171	70

3.3.1 Training experiments

The Dynamic Bayesian Classifier needs to be fed with some training data to create probability tables in order to correctly predict the locomotion task. This means that training experiments need to be carried out beforehand. In those experiments, each task was run separately to obtain a sufficient number of gait cycles. Given the selected terrain, this number of gait cycles was limited to 3 per task. The different tasks were recorded such that the subject had to mark a stop between each of them.

3.3.2 Testing experiments

After training, the algorithm performance can be evaluated as the number of misclassification errors occurring during a predefined path. This predefined path consisted in a circuit including a ramp, stairs and a flat surface. The ramp ascent task presents patterns that are more similar to level-ground walking. In Simon et al. [24], the ramp ascent task and level-ground walking are not well discriminated by the algorithm. Those tasks were therefore not differentiated by the algorithm and the performances of the algorithm were evaluated while taking this particularity into account. During the experiments, the number of steady-state steps and transitional steps were recorded in order to evaluate the performances of the algorithm in terms of steady-state and transitional errors. The circuit was defined in the environment depicted in Figure 3.3. Details about this circuit are given in Appendix B.

Figure 3.3: Experimental Setup

It should be noted that small alterations in the recorded signals were observed in a few cases. These alterations are probably due to the short range of the wireless router connecting the IMUs to the NGIMU Synchronised Network Manager. When the signal was only slightly altered (when only little data was missing), no changes were made to the signal as this kind of issue could also arise during daily activities performed by patients. Indeed, a model capable of handling such issues is considered more robust. However, when the misconnexion induced a total interruption of the signal for an extended period of the experiment, all the signals were truncated.

3.4 Signal processing

After recording the signals, multiple operations, as observable in Figure 3.4, were implemented. The python codes that were developed to implement each of these steps are attached to this document. The main function of each script is described in Appendix C. Each step will be described in the following sections and subsections. However, the synchronization and downsampling operations are linked to this particular work and are not necessary when working with sensors that are already synchronized and that have the same sampling frequency.

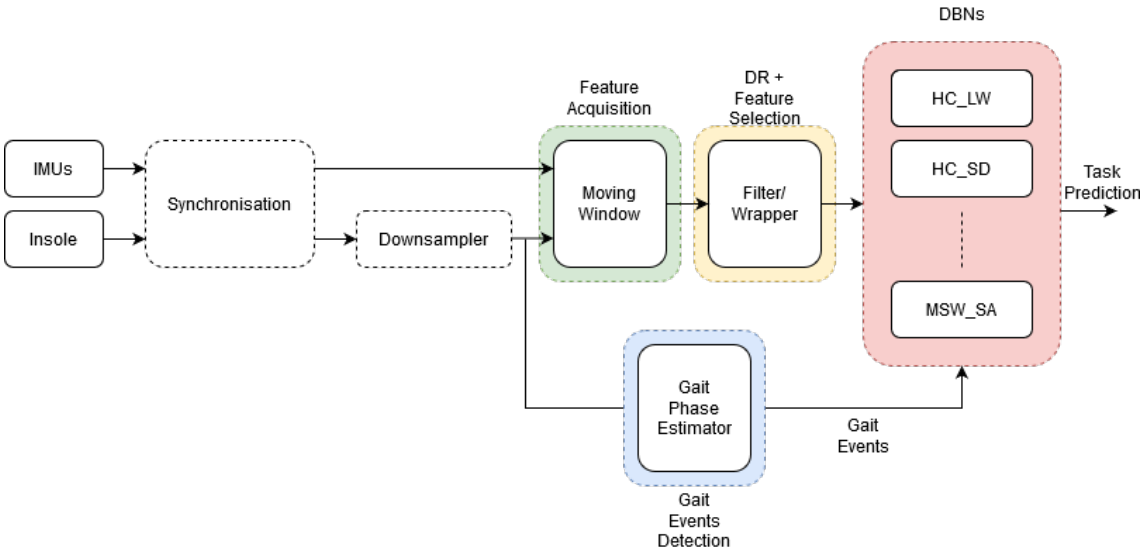


Figure 3.4: Block Diagram

3.4.1 Time offset adjustement

There was a noticeable offset between the recordings from IMUs and from the instrumented insole due to the small delay between each manual activation of the sensors. In the synchronization block, this delay was automatically removed by synchronizing the data with the system time. In Figure 3.5, the results of this synchronization operation are observable for a ramp ascent task performed at a gait cadence of 1.5 [Hz].

3.4.2 Downsampling operation

As the instrumented insole has a sampling frequency that is twice the sampling frequency of the IMUs, a downsampling operation, with a decimation factor of $M = 2$, had to be performed on the insole signal. Given the sampling theorem from Equation 3.1, which tends to prevent spectral aliasing,

$$f_s > 2 \cdot f_{max} \tag{3.1}$$

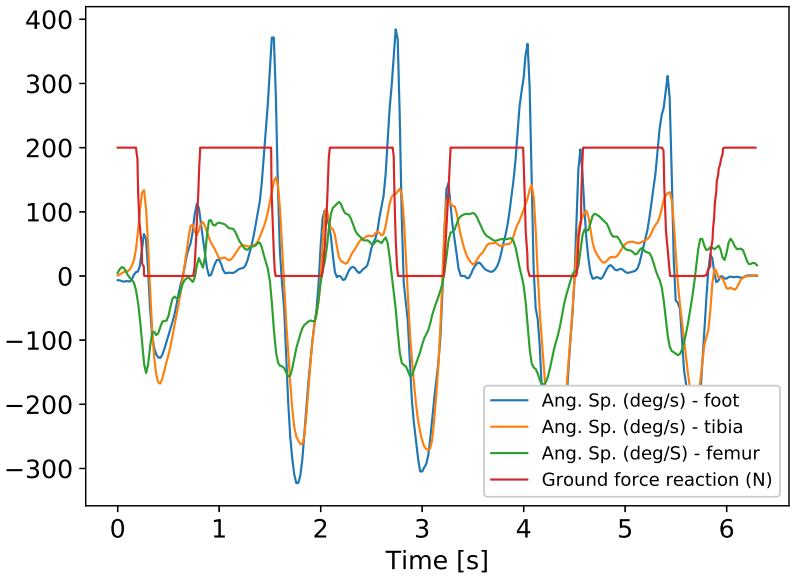


Figure 3.5: Synchronisation of a ramp ascent task performed by subject 1 at 1.5 [Hz]

where f_s is the sampling frequency and f_{max} is the maximal frequency component contained in the signal, it is understandable that this sampling rate conversion could introduce an aliasing effect if the signal $x[n]$ is not previously low-pass filtered as in Figure 3.6. Therefore, the filter, whose frequency response is given in Equation 3.2, limits the band

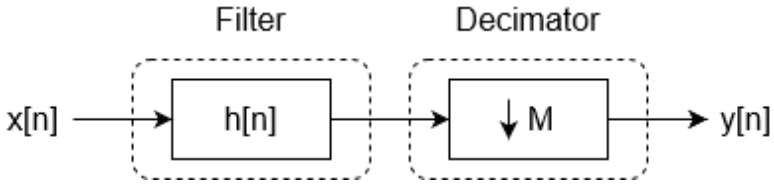


Figure 3.6: Downsampling operation

of the signal by a factor M [36].

$$H(e^{j\Omega}) = \begin{cases} 1 & |\Omega| \leq \pi/M \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

There are mainly 2 types of digital filters: finite impulse response digital filters (*fir*) and infinite impulse response digital filters (*iir*). As its name suggests, *fir* type filters output an impulse response with a finite period which is not the case for *iir* type filters. Both types have advantages and drawbacks but the main benefit for *fir* type filters is their linear phase property [37]. This implies that after passing into the filter, the signal does not suffer from any phase distortion. This is an important matter in this work as the phase of the ground force reaction signal, recorded thanks to instrumented insole, will be passed in a gait phase estimator in order to detect the different gait events. Therefore, the *fir* type filter will be preferred. To perform this operation, we used the python function `decimate` from the `scipy.signal` library.

3.5 Features Computation

In the following section, the terms variable and feature are used interchangeably.

3.5.1 Features acquisition

As in [24], the types of features computed for each recorded signal are the maximum, the minimum, the mean, the standard deviation, the initial and the final value of a particular time window. Those features are computationally quite inexpensive, which is important for a future real-time application of this work. For the training data, the features are computed on 300 [ms] sliding windows with an increment of 30 [ms] between each window. For the testing data, features are evaluated on 300 [ms] windows preceding each gait event for the non-delayed system and on 300 [ms] windows overlapping the gait events by 90 [ms] for the delayed system. The feature set contains 126 features, that is 6 features for each of the 21 recorded signals.

3.5.2 Dimensionality reduction

Dimensionality reduction (DR) is an important step while working with high dimensional data. Methods of dimensionality reduction will transform and project data into a new subspace. The objectives of such transformations are to be able to detect structures in data. Indeed, detecting structures allows to get information on the separability of data and to have a first intuition on the performances of a classifier. Moreover, DR enables to avoid the curse of dimensionality that is one of the major issues while working with machine learning techniques and high dimensional data. The curse of dimensionality is the fact that, as the number of variables (or dimensions) increases, the space of variables also increases. As a consequence, the amount of data necessary to fill this entire space increases. Therefore, even if it may seem, in theory, more efficient to have many variables, in practice that is not the case [38].

Two linear DR methods were considered: a principal component analysis (PCA) and a linear discriminant analysis (LDA). PCA aims at reducing the number of variables while keeping as much information as possible and consists in rotating the dataset from the original feature space until reaching the principal component space where each component accounts for the greatest possible variance in data. After performing this rotation, only the components associated with the largest variance are kept. In this new subspace, components are uncorrelated [38].

Contrary to the PCA, LDA is a supervised dimensionality reduction method, meaning that the label (task) associated with each sample is taken into account to determine the linear discriminants (the components) of the new feature space. It aims at maximizing distances between class means while minimizing the spreading within each class. LDA reduces the number of dimensions to K-1 dimensions (or less) where K is the number of classes [39]. It is important to make a clear distinction between LDA for classification (as presented in Chapter 3) and LDA for dimensionality reduction. They are based on the same mathematical concepts but do not follow the same objectives.

It may seem intuitive that LDA will outperform PCA, as it is a supervised dimensionality reduction method. However, for some applications, it has been shown that it is not always the case. PCA and LDA can also be used combined with each other [39].

3.5.3 Categorization

As DBNs are based on probability tables, they can only be built and perform inference on categorized data. Therefore, values within each feature have been binned while maximizing

the entropy binning of the initial sequence. Entropy based binning helps to categorize data by performing meaningful splits that retain most of the information about the original data. To perform these meaningful splits, we used the `entropy_based_binning` set of functions available on [40]. Before passing data into these functions, elements of each feature had to be mapped to the closest integers. Though necessary, it resulted in a loss of information. For each sequence passed as an argument, the `entropy_based_binning` functions generate all possible separations of the initial sequence into n non-empty bins, where n represents the number of categories and was set equal to 10 for each feature for implementation purposes. Each bin therefore contains a consecutive series of integers and no integer value can appear in two bins. The entropy corresponding to each separation is computed as

$$h(x) = -\sum_{i=1}^{n} p_i \cdot log_2 p_i \tag{3.3}$$

with p_i representing the probability density function of bin i (probability of being in bin i). The best separation or best binning is the one that produces the highest entropy. Since the maximal and minimal values, between which the binnings are determined for each feature, are taken from the training data, it is possible that some values within the testing set do not fall in the defined interval. Values exceeding the upper limit of the interval are therefore assigned to the category numbered n-1 and values bellow the lower limit are assigned the category numbered 0. A maximum of 10 categories per feature has been chosen to avoid too heavy computation.

3.5.4 Feature selection

For this subsection is is important to note that two approaches have been considered for the implementation of Dynamic Bayesian Networks. A first general DBN, allowing to discriminate among all classes, has been implemented. 7 mode-specific DBNs were also implemented but finally dropped. The graphs that are shown in this subsection correspond to the general classifier approach.

Feature selection is one of the core concepts of working with machine learning techniques. It can heavily impact the performance of a model. It does save some execution time by reducing the variables set and it reduces also the risk of overfitting. Overfitting is one of the major inconveniences in machine learning and must be avoided at all costs when building a model. It arises when a model has learned a particular set of data so well that this model is no longer useful to make predictions on a different yet similar set [41]. If the previously applied PCA and LDA pave the way for reducing the number of variables, using a lower number of features could be relevant for the classification task. Those features would have to be selected wisely however. This is where feature selection comes in. It should be noted that feature selection was only applied after transforming the data with a PCA alone. Indeed, as LDA reduces the feature subspace to 5 features only, no feature selection was performed after.

There are several techniques for carrying out a feature selection. Two types of approaches, in particular, can be distinguished: the filter approach and the wrapper approach. The first consists in assigning a relevance measure to each feature that is independent of the built model and then selecting the features whose measurement is higher than a fixed threshold. Unlike filters, wrappers assign relevance criterion to features considering the built models.

In this work, multiple features selection techniques were considered as it was very difficult to find a feature subset leading to good classification results. These different approaches

are presented in the following paragraphs.

Filter approach

Filters offer the advantage of being relatively faster than wrappers, in the sens that the set of selected features is obtained faster with filters. However, they are completely model-ignorant and the relevance criterion can sometimes be hard to estimate. This will be exposed in the following paragraphs [41].

Given that the information about the targeted tasks is available in the training set (supervised learning), the correlation between the features and the target can be determined. The correlation is restricted to [-1,1] with 0 indicating a complete decorrelation. One of the main drawbacks of a correlation measure is that it highlights only linear relations. Therefore a mutual information measure, that also identifies non-linear relationships, should usually be preferred. However, the correlation analysis generally gives a good indication as to which feature is most likely to be selected with another selection method. Figure 3.7 nevertheless shows a decorrelation between features that is due to the application of PCA. Indeed PCA maps the initial set of features to a reduced, decorrelated set (small degrees of correlation are however observable due to the categorization of features after applying the PCA). Stronger degrees of correlation between features and the target are observable in the last row or column of Figure 3.7.

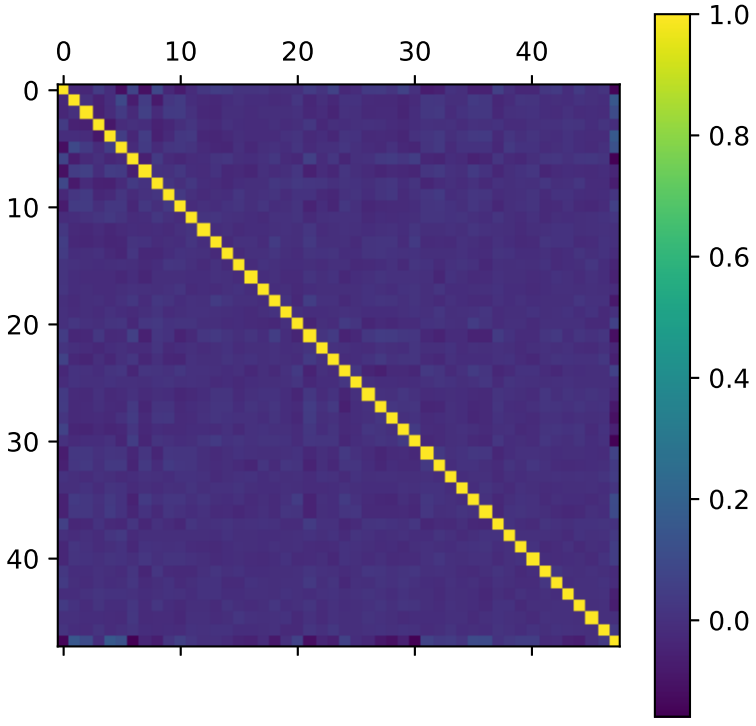


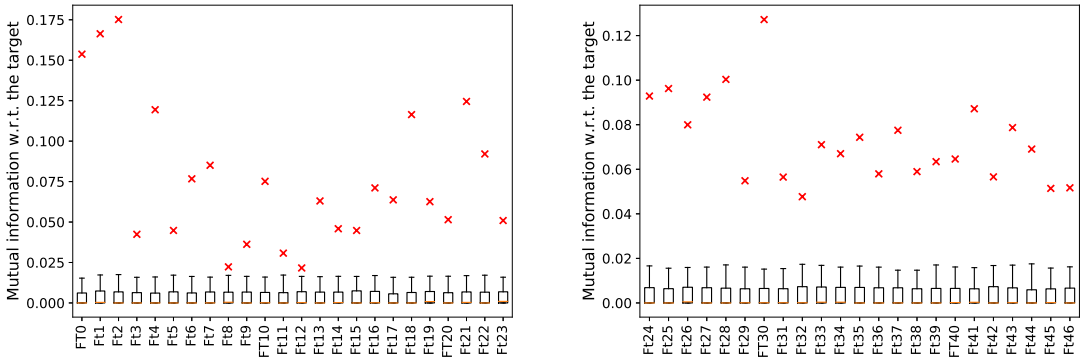
Figure 3.7: Correlation matrix presented as a colormap.

The mutual information $I(x, y)$ between two variables (x, y) indicates how the uncertainty

about one of the variables is reduced when the other variable is known. $I(x, y)$ can be computed as in 3.4 for discrete variables.

$$I(x, y) = H(x) - H(x|y) = H(y) - H(y|x) \tag{3.4}$$

In 3.4, $H(x)$ is the entropy of a variable which is a measure on its uncertainty. In this work, the mutual information was estimated thanks to the `feature_selection.mutual_info_classif` function from the `sklearn` library. The true mutual information is always non-negative but is not restricted to a particular range of values which makes it more difficult to interpret. Therefore, while using this measure as a selection criterion, a threshold that discriminates between relevant and non-relevant features should be established. To avoid choosing this threshold arbitrarily, a permutation test was used. This method was presented in [42] and consists in building Θ_i, the set of M mutual information measure of π_i where π_i is a random permutation of the particular feature x_i w.r.t. the target. This set has to be built for each feature. As no relation is expected between the random permutation of a feature and the target, the mean of each set Θ_i should be close to zero. The selected threshold for each feature is fixed as the 95th percentile of its corresponding Θ_i. Therefore, if the mutual information of a feature is greater than this threshold, one can consider with a significance level of 0.05 that this feature is in some way related to the target [43]. Figure 3.8 shows the results of the permutation test for the whole set of features where each feature was divided into 10 categories. The `feature_selection.mutual_info_classif` replaces negative estimates by zero which explains the half-boxplots. It is possible that for some features, the mutual information estimate (averaged over 100 estimations) falls below the 95th percentile threshold (indicated by the horizontal edge above the upper whisker). Those features could therefore be eliminated. However, by using the mutual information relevance criterion with the permutation test, all features are kept. Nevertheless, as an example the 6 features with the highest mutual information computed with this method could have been selected (see Chapter 4).



(a) (b)

Figure 3.8: Results of the permutation tests. Red crosses indicates the value of the mutual information averaged over 100 computations. The box-plots were drawn from 1000 random permutations of each feature.

The previous method does not take into account the fact that combined features can have a higher mutual information w.r.t. the target than if taken separately. Possible redundancies between features are also not considered. To overcome these issues, a forward greedy search has been implemented. This method will select features iteratively by evaluating

the mutual information of the group of features w.r.t. the target. The first selected feature is the one that yields the highest mutual information. The algorithm then tries each combinations with a second feature. The improvement significance of adding a new feature to the subset of selected features is assessed by applying a permutation test like before. However, Θ_i is only built on 100 random permutations for time-saving purposes. A feature is added to the subset if it passed the permutation test and if its addition to the subset produced the highest mutual information. Features are iteratively added according to this rule until no feature, when added, significantly increased the mutual information. The function `entropy_estimators.mi` from [44] is used to compute the mutual information of a subset of features w.r.t. the target. It should be noted that this estimator of the MI is not the same as the one used in the previous method. This explains the difference in MI estimates for the second feature in Figure 3.8 and in Figure 3.9. The features selected by this method are [2, 18, 1, 25, 10, 7] and the evolution of the MI of the subset w.r.t. the target while iteratively adding new features is shown in Figure 3.9.

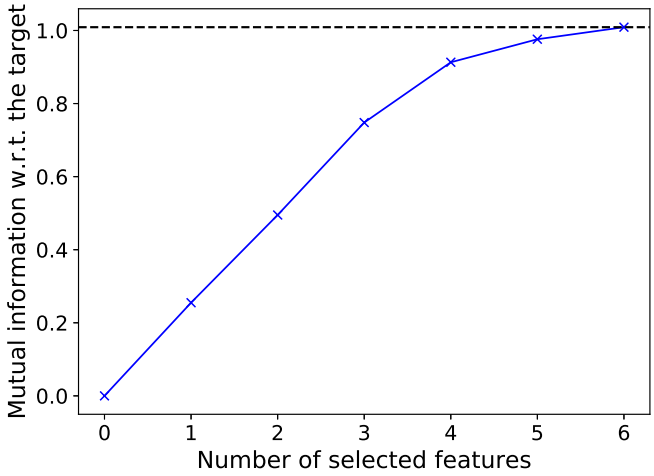


Figure 3.9: Evolution of the MI of the subset of features selected by the forward greedy search filter approach.

Wrapper approach

In the following paragraphs, the relevance criterion for a feature is assessed as the classification score obtained while using values of this feature for building the model and performing inference. However, since the task detection only occurs at specific gait events, the training set obtained from the training experiments could not be used. Indeed as explained in the following Section 3.6, a minimum number of steps is necessary to correctly detect gait events. For this reason, the 6 testing circuits will be divided into 3 training circuits (subject 1 at normal and fast gait cadences and subject 2 at slow gait cadence) and 3 testing circuits (subject 1 at slow gait cadence and subject 2 at normal and fast gait cadences). The model, though, was built on the initial training sets.

This second feature selection approach has also been implemented as part of this work. Wrappers, though time consuming, allow to easily determine the relevance criterion and will identify the optimal subset for the built model. However, the features selected by this method may not seem to be the most explanatory variables. It requires training the model and then assessing its performances on a new set of data [41]. The wrapper that

was implemented was based on a forward-backward greedy search. The algorithm starts with an empty set of selected features and will perform a forward search by iteratively adding features based on the model performance. It will first select the feature leading to the best performance and remove it from the remaining set of features. Afterwards, the algorithm will build the model on any possible combination of the previously selected feature with another feature coming from the remaining set of features. It will select the combination leading to the best performance on the testing data and so on. If the performance can not be further increased by adding a new feature, a backward search will be performed. This backward search will consider all possible removal of a feature from the features subset selected by the forward search and will remove a feature if it leads to the highest increase in performance. It will iteratively remove features until performance can not be further increased. The algorithm then comes back to its forward search and so on. It stops when the algorithm oscillates between forward and backward search without adding or removing new features. For each iteration of the forward-backward greedy search algorithm, the DBNs are built with the training data collected during the first phase of the experiments and tested on the dataset corresponding to some selected circuits. This method allows to consider whether some features may be redundant and therefore impact the model negatively.

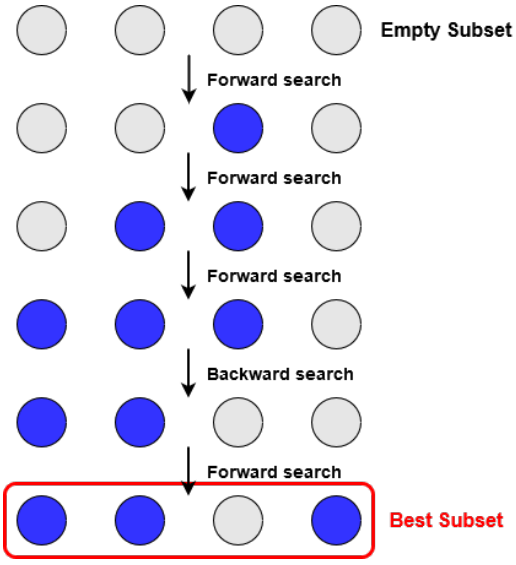


Figure 3.10: Example of feature selection while passing in the forward-backward wrapper.

This method was applied on the datasets of 3 training circuits. The circuits performed by subject 1 at slow gait cadence and by subject 2 at normal and fast gait cadence were left for testing. However, the collected datasets may be quite different as they come from different subjects walking at different gait cadences. Therefore, identifying the best features subset for a single dataset may be hard to generalize to others. To overcome this issue, only selected features that are common to a majority of datasets should be kept. However, no common features were selected by the forward-backward greedy wrapper and keeping them all also led to poor classification results. Therefore, this approach was dropped.

For that reason, another selection method was considered. A forward wrapper method was applied for each of the 3 training circuits datasets. This method returns the list of potential feature candidates sorted by decreasing score, the score being defined in Section

3.7.3. For each dataset, the feature in the first position is thus the feature leading to the best classification score. After running this method for each of the 3 datasets, the positions of each feature in the 3 lists that were produced were summed up and only features with the smallest position sums were selected to perform inference on the datasets of the testing circuit.

3.6 Gait event detection

A gait phase estimator similar to the one described by Yan et al. [45] was used to infer gait events. This gait event detection will allow to trigger a particular Dynamic Bayesian Classifier depending on the type of gait event that is detected.

The gait phase estimator built by the authors is composed of three subsystems: an event-detector, an AOs-based gait phase estimator and a phase error learning block. The architecture of such a system is observable in Figure 3.11.

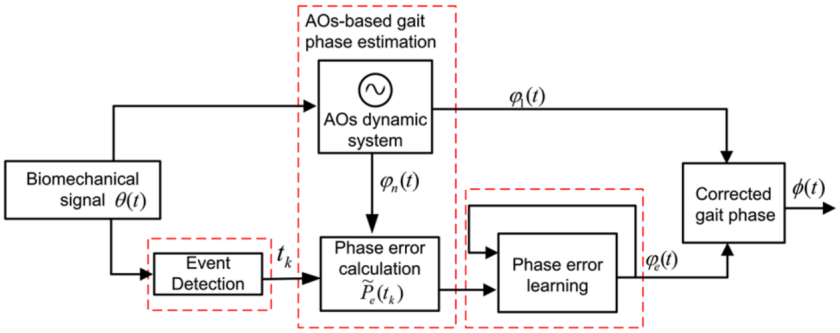


Figure 3.11: Architecture of the gait phase estimator. Each subsystem is surrounded by a dashed red line. From [45]

The study requires the tracking of a rhythmic locomotion signal. As the locomotion tasks investigated in this Master Thesis are periodic, one could link the start of each cycle to a particular event. In this work, the heel strike was selected as the event initiating a gait cycle. The event detector outputs the variable t_k, which represents the time where each new gait cycle is initialized and, therefore, the timing of this particular gait event. As explained in [45], false event detection can occur due to the smoothness of the biomechanical signal that is analysed, among others things. This paper introduces an alternative method by using adaptive oscillators that capture the frequency of an input signal and that allow to compute an estimate of the stride duration. As the investigation of locomotion tasks implies working with cyclic signals, events in two consecutive strides should be separated by an adaptative time interval depending on the stride period. The biomechanical signal that is passed in the 3 subsystems from Figure 3.11 is the magnitude of the vertical ground force reaction (vGFR), recorded with the instrumented insole. This signal was chosen because of its reliability, its consistency and the absence of signal alteration due to misconnexions during the experiments.

By supposing that an event, from stride k, is occurring at time t_k, the next detection of this event will arise at t*, which can be accounted for t_{k+1} only if it satisfies

$$t^* - t_k \geq \rho \cdot T(t^*) \Rightarrow t^* = t_{(k+1)} \tag{3.5}$$

where ρ is a constant in the range [0,1] (and was set to 0.68 to avoid missing a series of detections) and T(t) is the current stride period computed from $T(t) = \frac{2\pi}{\omega(t)}$, with $\omega(t)$ estimated from the adaptive oscillators subsystem.

In the second subsystem, a pool of adaptive oscillators (AO) is used to track a biomechanical signal and to produce a continuous phase variable that will be used to compute a phase error term. Adaptive oscillators are able to recover features from periodic signals such as phase, frequency, amplitude and offset in dedicated state variables by adapting their parameters. Figure 3.12 presents a diagram of this AOs subsystem. It decomposes the input signal into harmonics, each learned by a particular adaptive oscillator. All learned harmonics are then recombined to produce an estimation of the input signal. The difference between this estimation and the real input signal is used to correct the input of the AOs pool. Each adaptive oscillator is described by the following series of equations

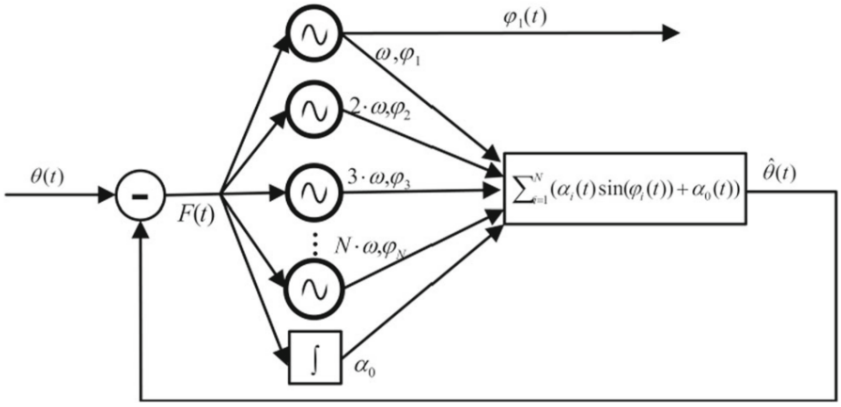


Figure 3.12: Diagram of the AOs subsytem. The input signal is decomposed into several harmonics, each of them learned by an adaptive oscillator in terms of phase, frequency and amplitude and then recombined to obtain an estimation of the input signal. A feedback loop allows to enhance this estimation. From [45]

$$\dot{\varphi}_i(t) = \omega(t) \cdot i + \nu_\varphi \frac{F(t)}{\sum \alpha_i} cos(\varphi_i(t)) \tag{3.6}$$

$$\dot{\omega}(t) = \nu_\omega \frac{F(t)}{\sum \alpha_i} cos(\varphi_1(t)) \tag{3.7}$$

$$\dot{\alpha_i}(t) = \eta F(t) sin(\varphi_i(t)) \tag{3.8}$$

$$\dot{\alpha_0}(t) = \eta F(t) \tag{3.9}$$

where ω is the fundamental frequency, φ and α are the phase and amplitude of the i^{th} oscillator, which learns the features of the i^{th} harmonic, and α_0 is a unique offset component, common to all oscillators. As indicated in Figure 3.12, F(t) is the error term between the estimated signal and the input signal. It can also be interesting to note that, instead of just synchronising its frequency to the one of the input signal, the state variable $\omega(t)$ integrates the phase update in order to learn the frequency of the input signal. The AO has the capacity to constantly adapt its frequency to the teaching signal and to memorize this input frequency. ν_φ, ν_{omega} and η are the learning parameters of the AOs pool. They can be tuned in order to adapt the learning speed of phase, frequency and amplitude. In [46], the authors give formulas to tune those parameters.

$$\eta = \frac{2}{\tau_\alpha}, \nu_\omega = \frac{20}{\tau_\omega^2} \text{ and } \nu_\varphi = \sqrt{24.2\nu_\omega} \tag{3.10}$$

where τ_α is the amplitude time constant and τ_ω is the frequency time constant. At the output of the AOs pool, the estimated signal is recomposed as

$$\hat{\theta}(t) = \sum_{i=1}^{N} \alpha_i(t) sin(\varphi_i(t)) + \alpha_0(t) \tag{3.11}$$

One of the main limitations of the method presented by Yan et al. [45] is the difficulty to fine-tune the parameters to efficiently track a particular biomechanical signal. Ronsse et al. [46] suggested that this fine-tuning is dependent on the type of biomechanical signal that is tracked, and that it should be performed by trial and error. The authors of [45] suggested using a pool of 5 AOs with $\nu_\varphi = 4$, $\nu_{omega} = 4$ and $\eta = 1.6$ for the tracking of vGFR. However, by doing so, the periodic signals should be synchronised within 7 strides. Proper fine-tuning of the parameters could lead to smaller delays but since the phase varies according to the gait cadences and that consequently the values of the parameters will be a compromise between the 3 gait cadences being investigated, some adaptive steps should be tolerated. Properly selecting those parameters requires in turn a trade-off between large values leading to a larger phase variability and small parameters, leading to a longer adaptation period to the new gait pattern. In this work, τ_α and τ_ω have been chosen as

$$\tau_\alpha = 2.0 \cdot \text{step\_dur}, \tau_\omega = 1.75 \cdot \text{step\_dur} \tag{3.12}$$

where step_dur is the step duration, depending on the gait cadence. Table 3.2 takes the values of the different parameters of the AOs pool according to the gait cadence.

Table 3.2: Parameter values of the pool of AOs.

Gait cadence	**0.8333 [Hz]**	**1.167 [Hz]**	**1.5 [Hz]**
Step duration	**1.2 [s]**	**0.857 [s]**	**0.667 [s]**
η	0.833	1.167	1.501
ν_α	10.476	14.669	18.876
ν_ω	4.535	8.892	14.723

From the pool of AOs, the fundamental phase φ_1 is learned and then normalized into the interval [0,2π] as in 3.13.

$$\varphi_n(t) = \text{mod}(\varphi_1(t), 2\pi) \tag{3.13}$$

A correct estimation of the gait phase should be fused with the gait cycle, therefore the gait phase should be equal to 0 at the start of each gait cycle, or in other words each time an event is detected at time t_k. A phase error term is computed as corresponding to the difference between 0 or 2π and the learned phase ϕ_n at time t_k.

$$P_e(t_k) = \begin{cases} -\varphi_n(t_k), \text{ if } 0 \leq \varphi_n(t_k) \leq \pi \\ 2\pi - \varphi_n(t_k), \text{ if } \pi \leq \varphi_n(t_k) \leq 2\pi \end{cases} \tag{3.14}$$

However, such conditions may lead to huge drops in this phase-error term when $\varphi_n(t_k)$ oscillates around π. The phase error is thus updated as

$$\tilde{P}_e(t_k) = \begin{cases} P_e(t_k) - 2\pi, \text{ if } P_e(t_k) > \frac{\pi}{2} \text{ and } \tilde{P}_e(t_{k-1}) < -\frac{\pi}{2} \\ P_e(t_k) + 2\pi, \text{ if } P_e(t_k) < -\frac{\pi}{2} \text{ and } \tilde{P}_e(t_{k-1}) > \frac{\pi}{2} \\ P_e(t_k) \text{ otherwise} \end{cases} \tag{3.15}$$

Based on these computations, the gait phase is corrected and estimated by

$$\phi(t) = \varphi_n(t) - \tilde{P}_e(t_k) \tag{3.16}$$

However, this method induces a discontinuity in $\phi(t)$ at each event detection. Therefore, the third subsystem is necessary to avoid such a discontinuity by implementing a phase error learning mechanism. A new state variable $\varphi_e(t)$ that learns the corrected phase error term within one stride is introduced. The dynamic of this new state variable is characterized by Equation 3.17.

$$\dot{\varphi}_e = \varepsilon_\varphi(t_k)\omega(t) \cdot e^{\omega(t)(t-t_k)} \tag{3.17}$$

where

$$\varepsilon_\varphi(t_k) = k_p[\tilde{P}_e(t_k) - \varphi_e(t_k)] \tag{3.18}$$

is the desired variation of $\varphi_e(t)$ within one stride cycle. Ideally, $\varphi_e(t)$ should converge to $\tilde{P}_e(t_k)$ during the cycle. k_p is a proportional gain and was set to 0.3 in the implementation whereas $\omega(t)$ is the estimated signal frequency from the AOs pool. Finally, the gait phase estimation is computed as the difference between the normalized gait phase and the phase error learning term. This difference is restricted between 0 and 2π.

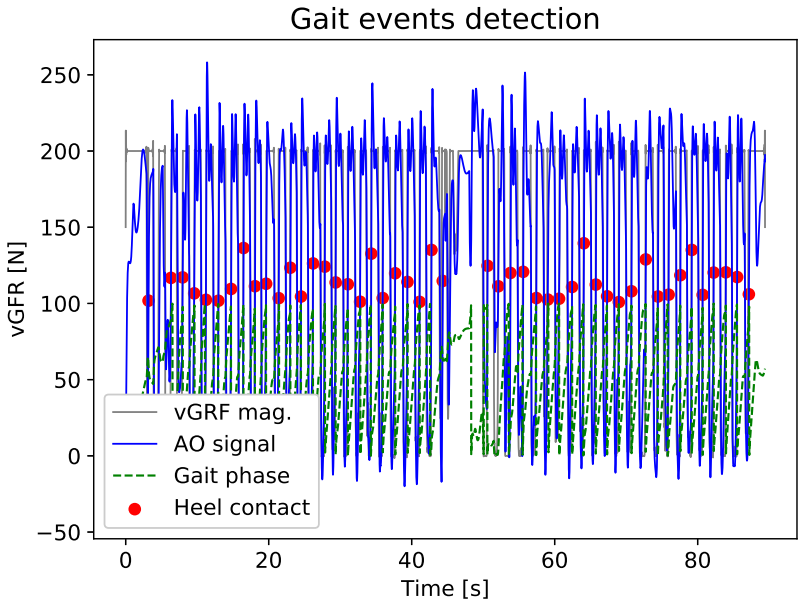
$$\phi(t) = \text{mod}(\phi_n(t) - \phi_e(t), 2\pi) \tag{3.19}$$

This final expression is divided by 2π to obtain a gait phase that varies between 0 and 1, and is then multiplied by 100 to be observable in Figure 3.13.

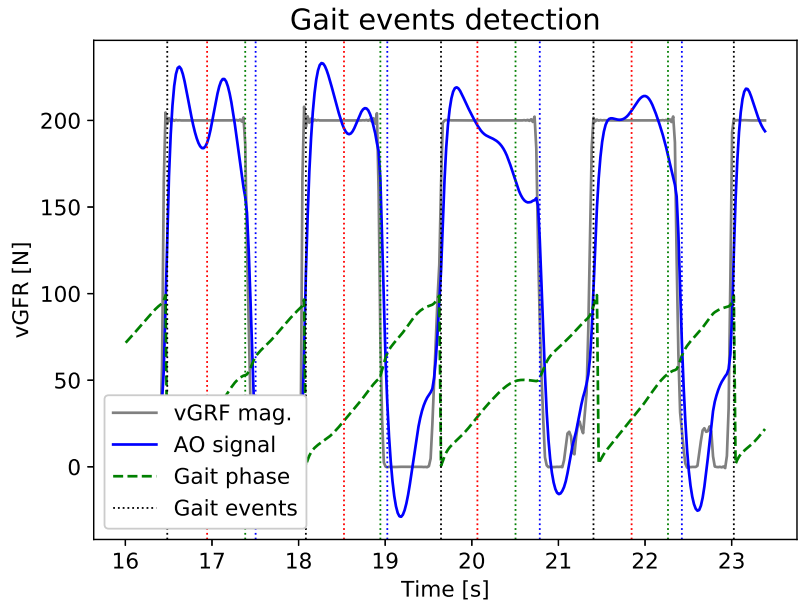
Figure 3.13 shows the result of the phase learning on the vGFR recorded while the second subject was performing the circuit at a normal cadence. Results of the phase learning for the other experiments are available in Appendix D. The number of adaptive steps varies between 2 and 4 with an exception of 8 adaptive steps for subject 1 performing the circuit at a slow gait cadence. The dashed lines indicated in Figure 3.13 reveal gait event detections. Given that heel contact was chosen as the start event for each gait cycle, mid-swing, toe-off and mid-stance should occur, at each stride, at the same percentage of the gait cycle. Mid-stance generally occurs at 30% of the gait cycle, toe-off is generally experienced at 60% of the gait cycle and mid-swing was defined at 75% [10]. Therefore, thanks to the information of the gait phase that constantly grows from 0 to 100 (the value achieved when the gait cycle is complete) detection of those events is possible.

All the equations in this subsection were extracted from [45] and [46]. The code needed to implement these 3 subsystems was largely provided by Virginie Otlet who used this gait phase estimator in her Master Thesis, submitted in 2019 [47]. However, this code has been translated from Matlab to Python and conditions have been added to detect the mid-stance events, the toe-off events and the mid-swings events. Moreover, the parameters of the AOs pool were fine-tuned for the purpose of this thesis.

Figure 3.13 illustrates the gait phase learning. The vGFR signal is the original ground reaction force, recorded from the instrumented insole. The AO signal is the output of the adaptive oscillators pool that is an estimation of the original signal. The gait phase in green is the corrected output phase of the AOs pool. In Figure 3.13, the gait phase learning is represented for subject 2 performing the circuit at normal gait cadence. Other experiments are shown in Appendix D. We can observe that the gait phase estimator sometimes fails to detect toe-off and mid-swing event. However, it performs well in detecting heel contact and mid-stance.



(a)



(b)

Figure 3.13: Gait phase learning for subject 2 performing the circuit at a normal gait cadence. Figure (a) shows the phase learning and heel contact detection over the entire circuit and Figure (b) highlights the phase adaptation and events detection during the first steps of the circuit. Heel contacts are represented by black dashed lines, mid-stance detection by red dashed lines, toe-off by green dashed lines and mid-swing by blue dashed lines.

3.7 Dynamic Bayesian Network Classifier

3.7.1 Model theory

Dynamic Bayesian Networks (DBN) extend the concept of Bayesian Networks by introducing time history information. They have been widely used in the context of locomotion intent recognition as suggested by the literature review from Chapter 2. As a reminder, in addition to the main study under investigation [24], A.J. Young used this concept in many of his articles about intent recognition both with mechanical sensors and neuromechanical fusion [6, 15, 22, 25]. L.J Hargrove also implemented a DBN to assess the contribution of targeted muscle re-innervation for EMG pattern recognition [27]. Finally, Spanias has also conducted a study based on EMG signals and a pool of DBNs [23].

At the core of the operations in a DBN are laws of probability and more particularly Bayes' Law. However, in order to understand the extended concept of DBN, a few paragraphs will be devoted to explaining how a Bayesian Network works.

Bayesian Networks

Bayesian Networks are a type of Probabilistic Graphical Model (PGM) that uses Bayesian inference for probability computation purposes [48]. Probabilistic Graphical Models exploit the structure of a distribution in order to describe it efficiently. They are based on a graphical representation of the problem whose nodes are the random variables and whose edges represent dependencies between those variables. Such structures can be learned from data or built manually. They are based on the property that variables tend to interact directly with very few others. Moreover, these structures are used to perform effective inferences and to compute the posterior probability of the output variable given the others [49].

Bayesian Networks are represented by directed acyclic graph (DAG) such as the one presented in Figure 3.14. This graph illustrates the Student example that is commonly used to explain the concepts relative to the Bayesian Network. This example is based on the problem faced by a company that wants to hire a graduating student. Simply put, this company will ideally want to hire an intelligent student but will have to take a decision based on a range of other information given that it is not possible to measure intelligence directly. It can consult the student's SAT score (score to the college admission test in the USA) and a letter of recommendation from his/her professor, for example. One assumption is that the quality of this recommendation letter is based only on the student's grades. This leaves us with the following dependencies: the SAT score depends only on the intelligence of the student, while the grades depend on both his/her intelligence and the level of difficulty of the class, and the quality of the recommendation letter depends only on the student's grades. It is based on this knowledge that the company should take its decision.

The graph in Figure 3.14 is directed in the sense that the edges, indicating dependencies between variables, have a direction. Variables in a BN are associated with a set of parents and a set of non-descendants. These concepts are important to understand the local Markov Property that is explained in the next paragraph. Parents of a variable are the variables on which it is dependent. For example, the grades obtained by the student in class are dependent on the level of difficulty of the class and on the intelligence of the student. Therefore, $\{Difficulty, Intelligence\}$ is the set of parents of *Grade*. The non-descendant set of *Grade* is the variables that are not descendants of this variable. In this example, this set is $\{SAT\}$.

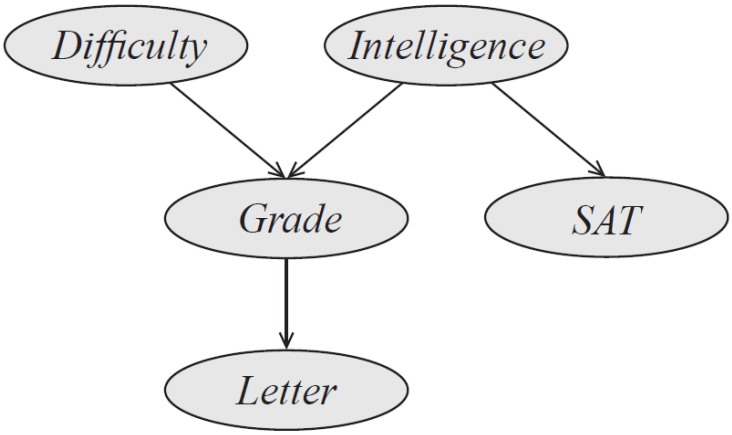


Figure 3.14: Structure of the Bayesian Network for the Student example. From [49]

In the following explanations, probabilities formulations and computations will be presented. Essentially, capital letters indicate random variables whereas lowercase letters indicate generic values. Letter X will be used to indicate a general random variable while explaining a general property. Notations T, F_1, F_2, ... F_n, on the other hand, indicate variables linked to this work problem.

Bayesian Networks satisfy the local Markov property. That property states that a node is conditionally independent of its non-descendants given its parents. The conditional independence property is at the core of the Naive Bayes Model, which is a very simple model that has been, however, widely used for classification purposes. A general example to explain this concept is given in [49]. Let us suppose that our student has previously taken the admission exams to get into two famous engineering schools. It can be reasonably supposed that succeeding in the first exam is not independent from succeeding in the second exam. Indeed, if we know that the student has been admitted in the first school, our estimate of the probability of his/her being admitted in the second school is higher since it works as an indication to his/her ability of successfully passing such exams. However, let us now suppose that both schools based their decisions only on the student's grades in one specific mathematics course. Based on these grades, the probability of being admitted to the first school should not influence the probability of being admitted to the second school. The grades already give that piece of information.

If two random variables X_1 and X_2 are conditionally independent given X_3, it can be written that

$$P(X_1, X_2|X_3) = P(X_1|X_3) \cdot P(X_2|X_3) \tag{3.20}$$

For random variables X_1, ..., X_n, the local Markov property is thus formulated as

$$P(X_1, ..., X_n) = \prod_{i=1}^{N} P(X_i|X_1, ..., X_{i-1}) = \prod_{i=1}^{N} P(X_i|Parents(X_i)) \tag{3.21}$$

by using the chain rule of probability.

Now that all these BN-related concepts have been exposed, the BN corresponding to this Master Thesis problem is presented in Figure 3.15. The n features represent those selected by the feature selection. The DAG presented in this Figure is rather simple but

more complex structures are also possible (such as the structure shown in Figure 3.14). The features are dependent on the task and therefore evidence about the features can be used to perform inference on the task.

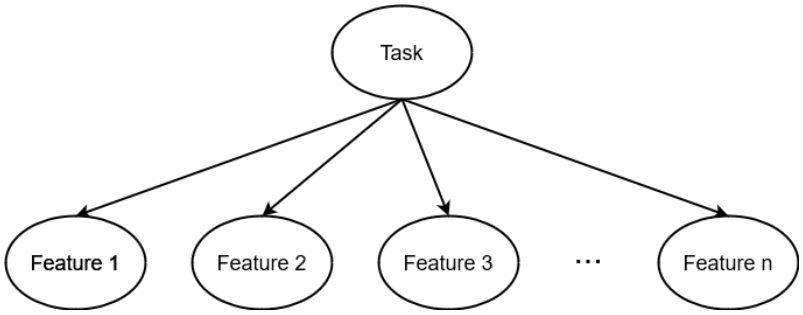


Figure 3.15: Structure of the Bayesian Network corresponding to this work problem

In this Master Thesis, a particular interest must be made in knowing the following conditional distribution

$$P(T|F_1, F_2, ..., F_n) \tag{3.22}$$

where T, F_1, F_2, ..., F_n are random variables representing the task that should be predicted, and the features that have been recorded and selected. All these random variables are categorical and y, f_1, f_2, ..., f_n will refer to the generic values of those random variables. It should be mentioned that the notation y, used to denote generic values of T, has been chosen to avoid any confusion with the temporal variable t.

The Bayes' rule can be used to express this conditional probability as

$$P(T|F_1, F_2, ..., F_n) = P(T|\bigcap_{i=1}^{n} F_i) = \frac{P(\bigcap_{i=1}^{n} F_i|T) \cdot P(T)}{P(\bigcap_{i=1}^{n} F_i)} \tag{3.23}$$

The direct application of such a formulation is intractable, especially when the number of variables grows. Indeed, the amount of data necessary to correctly estimate the term $P(\bigcap_{i=1}^{n} F_i|T)$ exponentially grows with n. The local Markov Property and the conditional independence assumption between the features when knowing the target will help to overcome this issue. Thanks to this strong yet useful assumption, Equation 3.23 can be rewritten as follows

$$P(T|\bigcap_{i=1}^{n} F_i) = \frac{\prod_{i=1}^{n} P(F_i|T) \cdot P(T)}{P(\bigcap_{i=1}^{n} F_i)} \tag{3.24}$$

By also taking the independence assumption between features, the joint probability $P(\bigcap_{i=1}^{n} F_i)$ can be factored as $\prod_{i=1}^{n} P(F_i)$. However, this term is just a normalization term and will not be taken into account in the implementation.

Equation 3.24 will be at the core of the task classification algorithm. However, this equation only deals with information coming from one time step. DBNs allow to integrate past information in the probabilities computation.

Dynamic Bayesian Network

DBNs are composed of static Bayesian Networks modeled over an arrangement of time-series. DBNs are also called N-timeslices Bayesian Networks. This name highlights that in these networks the value of a variable at time T can be calculated from prior and current values. Each timeslice is conditionally dependent on the N-1 previous ones.

In this work, a two-timeslice Bayesian Network was used. Figure 3.16a shows an unfolded version of this DBN. A common and more compact representation is given in Figure 3.16b. The red arrow symbolizes the conditional dependence of the current task on the previous one. The red frame with the number 1 in Figure 3.16b indicates the number of time transitions implicated in that dependence.

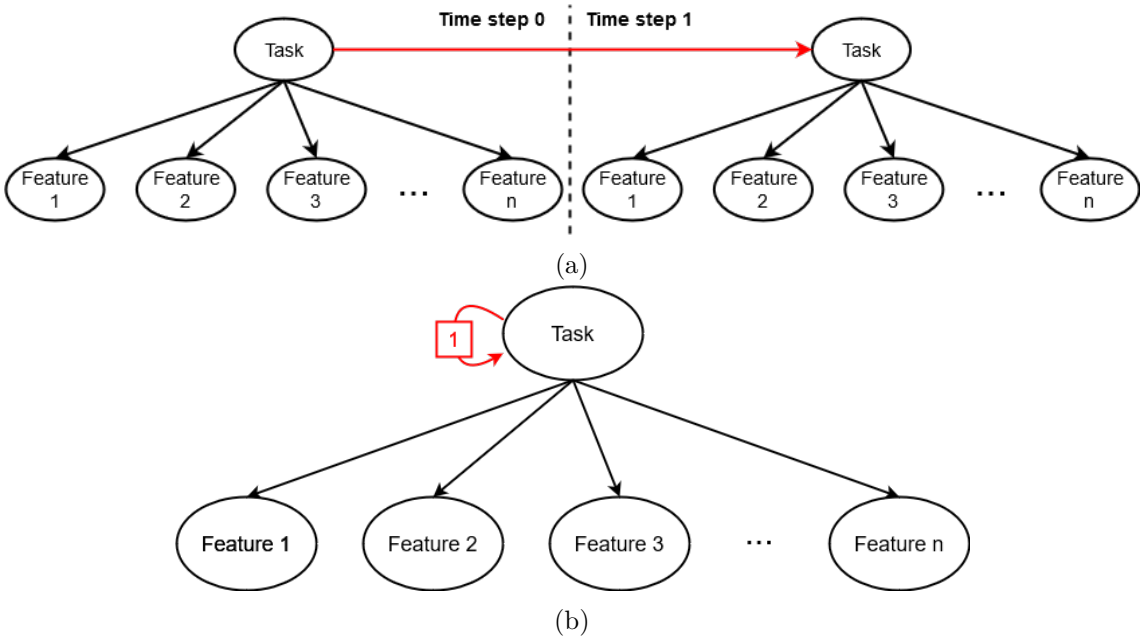


Figure 3.16: Structure of the Dynamic Bayesian Network corresponding to this work problem. In (a), the dashed line indicates that the two structures are coming from different time steps. The red arrow indicates direct dependency of the Task at time step 1 on the Task at time step 0. (b) is the folded representation of (a).

DBNs use the previous equations to calculate the maximum a posteriori (MAP) estimate which corresponds to the task (the class) with maximum posterior probability. The MAP estimate is denoted $\hat{T}_{MAP}$ and is computed in Equation

$$\hat{T}_{MAP} = \text{argmax}(P(T|\bigcap_{i=1}^{n} F_i)) = \text{argmax}(\frac{P(\bigcap_{i=1}^{n} F_i|T) \cdot P(T)}{P(\bigcap_{i=1}^{n} F_i)}) \tag{3.25}$$

where $\bigcap_{i=1}^{n} F_i$ is the set of selected features and P(T) is the prior probability. The prior probability is computed as a multiplication of the previous step posterior probability $P(T|\bigcap_{i=1}^{n} F_i)_{t-1}$ by the transition matrix ϕ.

$$P(T)_t = P(T|\bigcap_{i=1}^{n} F_i)_{t-1} \cdot \phi \tag{3.26}$$

The transition matrix ϕ, given in Table 3.3, is the matrix of probabilities of transitioning from one task (on columns) to another (on rows) and has been built by avoiding mode-transitions that were really unlikely to occur during a daily activity. Besides, those tasks could also be prevented from occurring by the choice of DBNs (see Table 3.4). The transition probabilities were not inferred from data as the number of level-ground walking steps are too great. However, these probabilities were inspired from data and based on intuition. Indeed, it seems logical to think that when a subject is performing one particular locomotion task, there is a higher probability that he will still be performing that task on the next step rather than transitioning to another locomotion task. For example, it can be understood from Table 3.3 that there is a 60% chance of transitioning from a level-ground walking step to a level-ground walking step against a 10% chance of transitioning from a level-ground walking step to a stair ascent step. Besides, transitions from standing mode have been weighted differently because it is less common for an individual to stay standing for a long time period without deambulating. All these probabilities have been inferred from personal intuition and observations and can be re-weighted for specific applications.

Table 3.3: Transition matrix ϕ

	LW	**SA**	**SD**	**RD**	**ST**
LW	0.6	0.2	0.2	0.2	0.15
SA	0.1	0.6	0	0	0.15
SD	0.1	0	0.6	0	0.15
RD	0.1	0	0	0.6	0.15
ST	0.1	0.2	0.2	0.2	0.4

In the following equations, the indexes 0 and 1 indicate that evidence of a random variable was observed at the previous timeslice or is observed at the current one. In light of this new structure, the previously stated equation that will be used to perform inference was updated as follows

$$P(T_1|\bigcap_{i=1}^{n} F_{i1}) = \frac{\prod_{i=1}^{N} P(F_{i1}|T_1) \cdot P(T_1)}{P(\bigcap_{i=1}^{n} F_{i1})} \tag{3.27}$$

where $P(T_1)$ is directly dependent on the previous task and is computed as

$$P(T_1) = P(T_0|\bigcap_{i=1}^{n} F_{i0}) \cdot \phi \tag{3.28}$$

For the implementation of DBNs, two approaches were considered. First, inspired from the experiments implemented in [24], the article presented in Chapter 2, multiple mode-specific classifiers were trained and used for inference. Some of these classifiers were adapted or removed to take every possible transitions in the testing circuits into consideration. In this implementation, only 7 classifiers, observable in Table 3.4, were implemented. However, such classifiers had to be trained on circuit datasets in order to be able to detect gait events while building those new training datasets. Besides this, original training datasets could not be used due the minimum number of steps necessary to synchronise the AOs pool from the gait phase estimator. Such datasets were built on data from 5 of the 6 available circuit experiments. The circuit performed by subject 1 at fast gait cadence was left for testing purposes.

Table 3.4: Mode-specific classifiers. Adapted Table from [24]. LW stands for level-ground walking mode, RD for ramp descent, SD for stair descent, SA for stair ascent and ST for standing mode. The red elements are those that have been modified from the previously presented Table 2.7.

Gait Phase	Classifier	Active in	Description	Number of Classes
Heel Contact	HC_LW	Level-ground walking	Predicted steady-state and the transitions from level-ground walking to stair and ramp descent	3 (LW, SD, RD)
	HC_SD	Stair descent	Predicted steady-state and the transitions from stair descent to level-ground walking	2 (LW, SD)
	HC_RD	Ramp descent	Predicted steady-state and the transitions from ramp descent to level-ground walking	2 (LW, RD)
	HC_ST	Standing	Predicted steady-state and the transitions from standing to level-ground walking	2 (LW, ST)
Mid-Stance	MST_LW	Level-ground walking	Predicted steady-state and the transitions from level-ground walking to standing	2 (LW, ST)
Toe Off	TO	Level-ground walking, ramp descent, stair descent	Predicted steady-state and the transitions from level-ground walking to stair ascent and between level-ground walking rand ramp/stair descent	4 (LW, SA, SD, RD)
Mid-Swing	MSW_SA	Stair ascent	Predicted steady-state and the transitions from stair ascent to level-ground walking	2 (LW, SA)

From Table 3.4, it appears that none of these classifiers allowed to discriminate the ramp ascent task as it was considered to be too similar to the level-ground walking task. This approach was expected to produce very poor classification results due to the small number of training data that could be used to implement it. Indeed, feature selection and models based on such small training datasets were really hard to generalize. Therefore a second approach was considered. A general DBN was used that allowed to discriminate among all 5 possible tasks (while excluding ramp ascent). This DBN was not mode-specific and was trained with the data from training experiments and tested on the circuits. This approach contradicts what has been shown in the literature as the first approach has proved to yield better results [18, 23]. However, given the small number of experiments performed, this was deemed to be a good compromise.

3.7.2 Implementation

As the structure of the DBN used in this Master Thesis was quite simple, no library was used for its implementation. However, it should be noted that the following functions are not generalizable to more complicated structures.

A first function `DBNetwork` is explained in details in Algorithm 1. It takes as `Inputs` the training data, including only the selected features and the targeted tasks, the tasks that the classifier should differentiate and the number of categories (`ncategories`) per feature. The `Output` is the model in the form of a conditional probability table (`ntasks` x `ncategories` $^{\texttt{nfeatures}}$), containing the probabilities of having a particular set of features when knowing the task. Each element of this conditional probability table is computed as the product of each $P(F_i = f_i|T)$ [l.15] as stated in equations 3.24 and 3.27, calculated from a pivot table constructed from the training data. Pivot tables are tables that summarize more complex data tables into relevant statistical elements. For each feature, the pivot table contained the number of elements in the training set corresponding to a particular generic value when the observation was classified as a specific task. From such tables, values of $P(F_i = f_i|T)$ could easily be retrieved. However, a frequently asked question while working with insufficiently large datasets is what happens if none of the training examples has the generic value f_i for a given target value ? As $P(\bigcap_{i=1}^{n} F_i|T)$ is computed as the product of probabilities of evidence of a feature given the target, it will immediately bring that value down to 0. An alternative would be to consider smoothing

the estimate $P(F_i = f_i|T)$ [50] by

$$P(F_i = f_i|T) \leftarrow \frac{n_{f_i,y} + \gamma p_{f_i,y}}{n_y + \gamma} \tag{3.29}$$

where

- $n_{f_i,y}$ is the number of training examples with $F_i = f_i$ and a task label y
- n_y is the number of training examples with task label y
- $p_{f_i,y}$ is the prior estimate for $P(F_i|T)$ (and will typically be set to 1/ncategories)
- γ is the weight given to $p_{f_i,y}$

Algorithm 1: DBNetwork

Inputs : data: dataset including only the columns from the original dataset corresponding to the selected features and a column for the target,
ncategories: the number of categories per features,
discr_tasks: tasks that the classifier should discriminate among

Output: CPT_T: model in the form of a conditional probability table

```
% Separate features and target, nfeatures is the number of selected features:
features ← np.delete(data, nfeatures)% remove last column of data, containing the target
target ← data[:, nfeatures]
% Categorize tasks:
target ← categorize_tasks(target, discr_tasks) % target vector where each task has been associated with a number between 0 and 4 depending on what is in discr_tasks
% Conditional probability tables, dictionary:
for each selected feature i do
    CPT ← CPT.update("CPT_FTi": cpt_i) % cpt_i from the pivot tables
end
% 5 possible tasks: LW, SA, SD, RD, ST
for each possible task i do
    for j in range ncategories^nfeatures do
        index ← base_convert(j, ncategories) % convert j into base ncategories numbering, yielding "000000", "000001", ... "000011", "000012", ... values
        for k in range nfeatures do
            CPT_T[i][j] ← CPT_T[i][j] · CPT_{k[i][index[k]]}% product of each feature corresponding conditional probability
        end
    end
end
return CPT_T % model final probability table where each column has a sum of probability equal to 1
```

A second function `makeInference` takes as `Input` the model on which the inference should be performed, the posterior probabilities at the previous time step, the features computed at the current time step from the testing data and the number of categories per feature. The `Output` are the vector of posterior probabilities at time step t and the MAP task. In this function, knowing the values of the features computed at time t and the model on which inference should be performed, $P(\bigcap_{i=1}^{n} F_i|T)$, the likelihood probability is obtained by simply identifying the correct column in the conditional probability table of the model `[l.1-4]`. The prior probability is computed from the previous posterior probability vector

and the transition matrix [l.6] that is assigned in this function, as in Equation 3.28. Finally the posterior probability vector at the current time step is computed by performing an element-wise multiplication between the likelihood probability vector and the prior probability vector [l.7], as stated in Equation 3.27. The normalization term $P(\bigcap_{i=1}^{n} F_i)$ was not computed. The MAP task is identified as the one corresponding to the maximum posterior probability at the current time step [l.10].

An issue to be addressed is the fact that values in the posterior probability vector become smaller and smaller at each call of the `makeInference` function and quickly converge to 0. This is due to the small number of training data versus sometimes a relatively high number of features and of categories per feature, necessary to correctly discriminate among tasks. To overcome this problem, at each call of the `makeInference` function, the posterior probability is computed as the sum of the log likelihood with the log of the prior [l.7]. This is done instead of performing an element wise multiplication.

Algorithm 2: makeInference

Inputs : model: tuple containing the general conditional probability table and the tasks that the classifier should discriminate among,
prev_posterior: posterior probability vector from the previous time step,
data: features computed at the current time step,
nb_categories: nb of categories per feature

Output: posterior: posterior probability vector from the current time step,
map_task: maximum a posteriori task from the current time step

```
% Recover the column of the model CPT from features values:
index ← toDeci(str_data, nb_categories) % str_data is the string containing the features values, convert into base 10 numbering
% Likelihood:
lkli_proba ← model[0][:, index]
% Prior:
prior ← phi.T@prev_posterior % matrix multiplication between transposed transition matrix and the previous posterior probabilities
posterior ← np.log(lkli_proba)+np.log(prior) % sum of the log-likelihood and the log prior, the logarithmic base is used to avoid a quick convergence to 0
posterior ← np.exp(posterior) % re-switch to a non-logarithmic base
discr_tasks ← model[1]
map_task ← np.arg_sort(posterior[discr_tasks])[-1] % return the index that has the highest posterior probability among the indexes contained in discr_tasks
return posterior, map_task
```

3.7.3 Performance assessment

To evaluate the performances of these DBNs, two main scoring functions were considered. The first one is a scoring function that balances the possible occurrences of the different tasks. This function is implemented in the `scikit-learn` library and is called the `balanced_accuracy_score` [51]. It allows to deal with imbalanced datasets as it is the case in this work. Indeed, as previously explained, the circuits include many more level-ground walking steps than any other tasks. The score was computed as

$$\texttt{balanced-accuracy}(y, \hat{y}, w) = \frac{1}{\sum \hat{w}_i} \sum_i \mathbf{1}(\hat{y}_i = y_i)\hat{w}_i \tag{3.30}$$

where

- y_i is the true value of the i-th sample

- $\hat{y}_i$ is the predicted value of the i-th sample
- $\mathbf{1}$(x) is the indicator function whose value is simply 1 when $\hat{y}_i = y_i$ and 0 otherwise
- $\hat{w}_i$ is the adjusted sample weight and is computed in the following equation

$$\hat{w_i} = \frac{w_i}{\sum_j \mathbf{1}(y_j = y_i) w_j} \tag{3.31}$$

with j indicating a locomotion task and w_i set to 1 for all samples.

With this function, the classifier can not show performance above chance simply by taking advantage of imbalanced datasets, since the performance will be dropped to $\frac{1}{\text{nb\_tasks}}$ in that case.

However, one could argue that individuals generally walk more than they perform ramp descent or stair ascent tasks. Therefore, this raises the question of balancing the score according to the probability of daily life occurrence of tasks. It seems unfair to harshly penalize a classifier that predicts the level-ground walking task most of the time when this task has a greater likelihood of occurring. Based on this intuition, a second scoring function was implemented. It should be noted that the terms in this equation are not always equivalent to the previously presented ones.

$$\texttt{intuitive-accuracy}(y, \hat{y}, w) = \sum_i \mathbf{1}(\hat{y}_i = y_i) w_i \tag{3.32}$$

where

$$w_i = \sum_j \frac{w_j}{\mathbf{1}(y_j = y_i)} \tag{3.33}$$

Here, w_j is the probability attributed to the locomotion task. Such probabilities are summarized in the following expressions

$$P(LW) = 0.6, P(SA) = 0.1, P(SD) = 0.1, P(RD) = 0.1, P(ST) = 0.1 \tag{3.34}$$

Those probabilities are, again, based on intuition but they can easily be adapted if other approaches are considered.

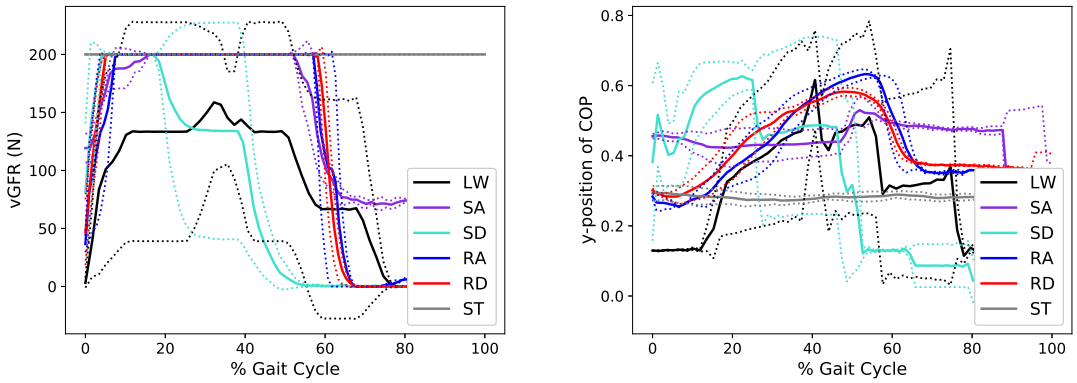
Chapter 4

Results

This chapter exposes the classification results obtained with the general DBN presented in the previous chapter, for the non-delayed and delayed approaches. It also presents the computational time required by the algorithm for inference. Some results for the mode-specific DBNs approach are presented in Appendix E.

4.1 Raw data

Figure 4.1 shows the evolution of signals during a gait cycle for each task. Singular patterns as in Figure 4.1a are due to the difficulty for the subject to keep a constant gait cadence over the entire experiment, despite prior training. In these figures, even if general patterns are sometimes noticeable, high standard deviations indicate that these signals may considerably vary during the course of the experiments, making it hard to identify a specific pattern corresponding to a particular locomotion task. For example, the ramp descent task seems to be characterized by a leg angle velocity that is faster (in absolute value) than for the other tasks in the first moments of the stance phase 4.1e. However, the high standard deviations observable for ramp ascent and level-ground walking tasks indicate that such tasks may also have fast leg angle velocity. The classifiers would then not differentiate those tasks from a ramp descent task. Moreover, some signals such as the foot, leg and thigh acceleration in the sagittal plan do not seem helpful for classification.



(a) (b)

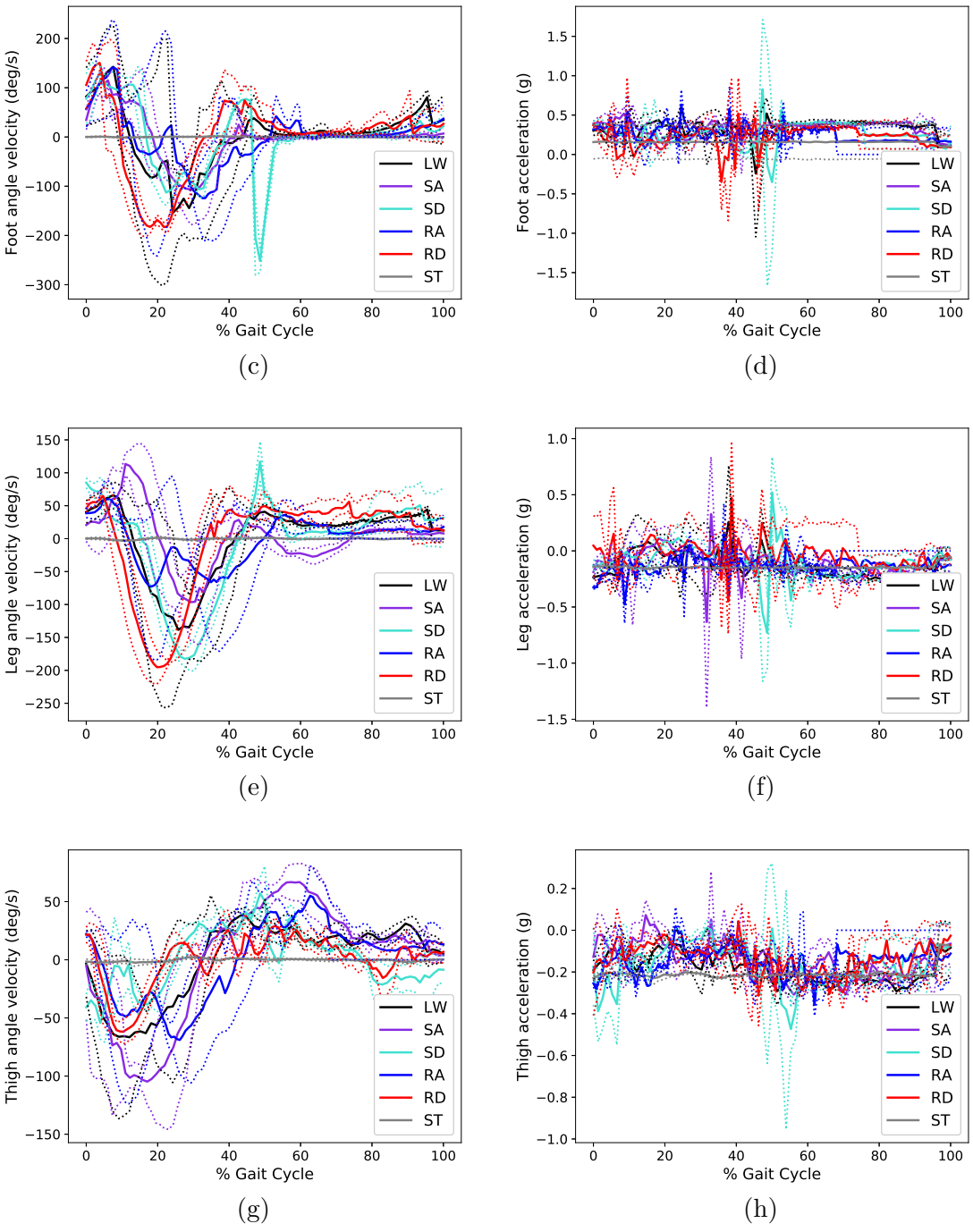


Figure 4.1: Signal patterns from the vertical ground force reaction, y-position of the center of pressure, as well as sagittal gyroscope and accelerometer for the foot, the leg and the thigh are shown for the 6 locomotion modes. Those modes are level-ground walking (black), stair ascent (purple), stair descent (turquoise), ramp ascent (red), ramp descent (blue) and standing (gray). Data is averaged over the gait cycles performed by subject 1 at slow gait cadence and normalized over 1 stride. 3 strides were avegared for each task except for the ramp ascent were only 2 strides were averaged due to a lack of data. Standard deviations across strides are shown by dashed lines.

As a reminder, the features computed from each signal are the mean, the standard deviation, the maximum and minimum values and the initial and final values, taken from 300 [ms] windows. From Figure 4.1, a possible intuition is that those features would not really help to discriminate among the investigated tasks. Figure 4.2 shows these features computed from the signal of the sagittal leg gyroscope.

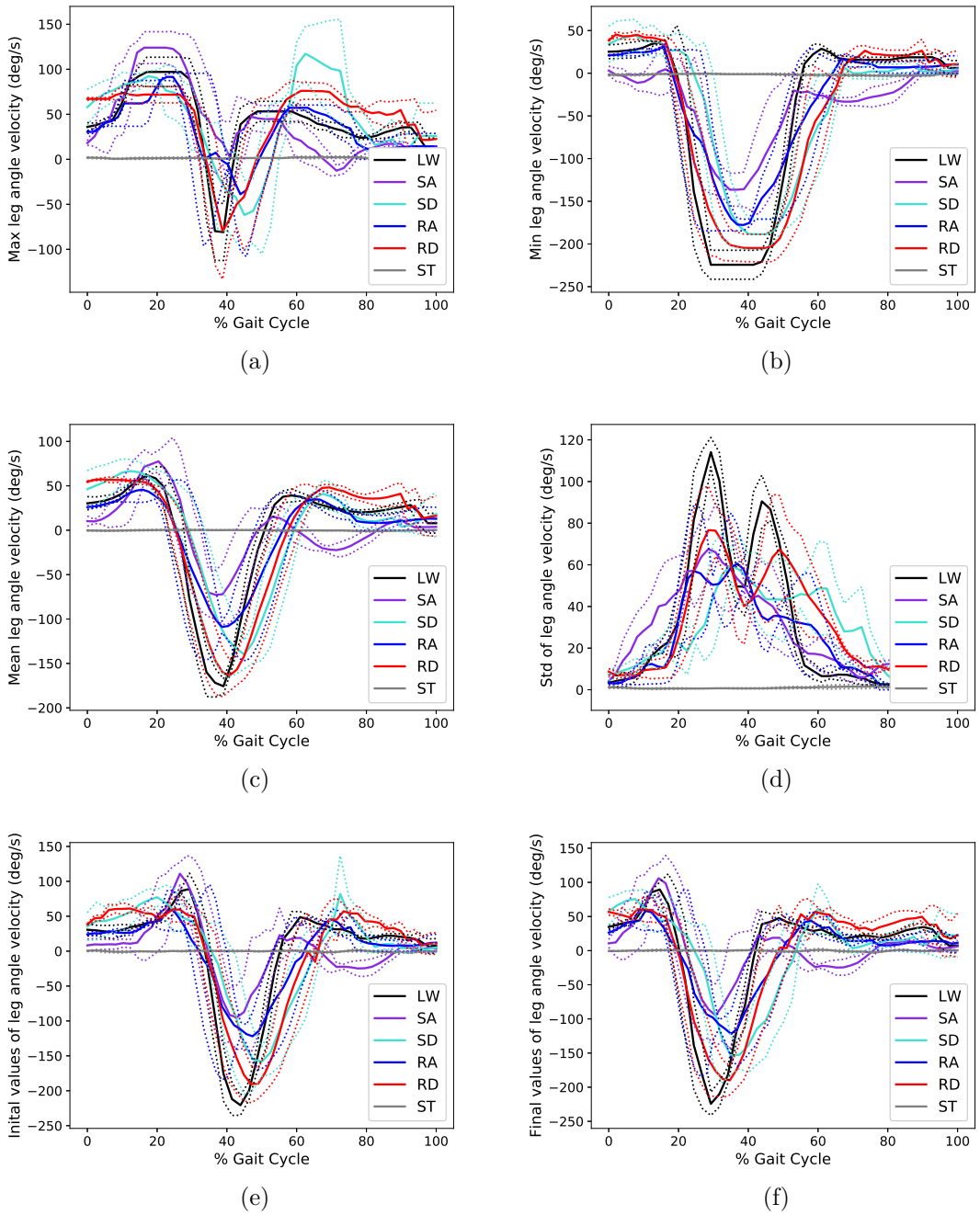
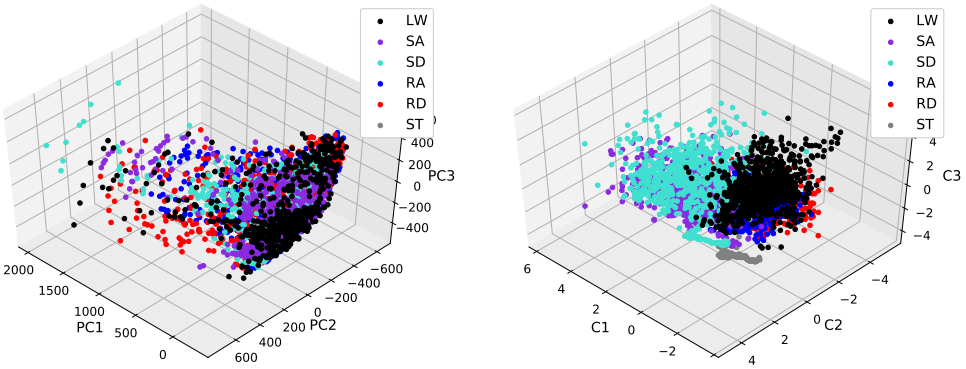


Figure 4.2: The features computed from the sagittal leg gyroscope are averaged over 3 gait cycles for each task (except for ramp ascent task). The features are computed over 300 [ms] windows taken from the original, downsampled signals. Maximum (a), minimum (b), mean (c), standard deviation (d), initial (e) and final (f) values are shown.

Figure 4.2 supports the previous intuition in the sense that ranges $[\mu - \sigma,\ \mu + \sigma]$ still highly overlap between different locomotion tasks. It should however be noted that there is a difference between the average values for different tasks and for particular features (for example, the mean leg angle velocity shows a difference in average among tasks). Any conclusions about Figures 4.1 and 4.2 should be drawn carefully due to the small number of gait cycles used for the computation of averages and standard deviations for those figures.

4.2 Algorithm performances

After computing all the features, a dimensionality reduction method was applied. Results of the PCA and the LDA are shown in Figure 4.3. Better results can be expected with the LDA because it takes the targets into account in its process for dimensionality reduction. This becomes clear from observing Figure 4.3a and Figure 4.3b. The LDA method was therefore selected to reduce the dimensionality leading to a feature subspace of 5 components. In the following figures, the ramp ascent task is considered separately from the level-ground task in order to get a complete view of classes separation. However, while building the model, ramp ascent tasks where re-labelled as level-ground walking.



(a) (b)

Figure 4.3: Training data along the 3 first components obtained through PCA (a) and LDA (b)

After applying the dimensionality reduction, features should be categorized in order to build DBNs. This resulted in a categorization into 10 categories as shown in Figure 4.4. One can observe that the entropy-based binning preserves the information well and that classes are still well separated after the categorization.

When performing inferences on the general model that was trained with data from Figure 4.4, poor classification results were obtained. Those results, summarized in Table 4.1, are averaged over the 6 testing circuits.

Table 4.1: Classification results obtained with the general DBN after applying an LDA dimensionality reduction.

`balanced-accuracy`	21.163 ± 4.810 %
`intuitive-accuracy`	14.572 ± 5.121 %

In light of these results, two options were considered: applying a combination of PCA and LDA for dimensionality reduction, and applying only PCA for dimensionality reduction

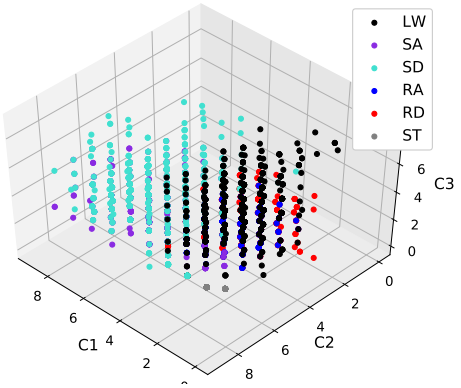
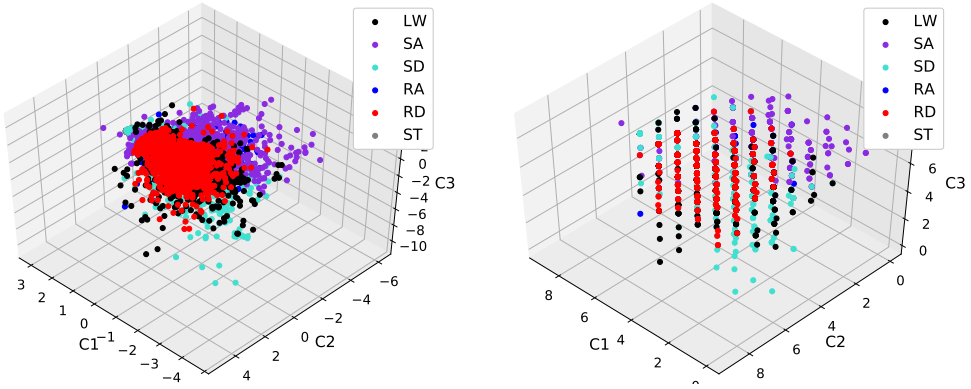


Figure 4.4: Categorized training data along the 3 first components obtained through LDA

but performing a subsequent feature selection. Training data for the first approach is shown in Figure 4.5.



(a) (b)

Figure 4.5: Training data along the 3 first components obtained through successive PCA and LDA transformations (a) and same data after applying categorization (b)

A separation between classes is observable even if this separation seems less well-defined than with an LDA transformation only. This approach however shows a net increase in classification performance (Table 4.2). These differences in classification results between LDA and PCA+LDA for dimensionality reduction will be discussed in the following chapter.

The following table shows the classification results obtained after applying successive PCA and LDA for dimensionality reduction.

Table 4.2: Classification results obtained with the general DBN after applying PCA+LDA dimensionality reduction

`balanced-accuracy`	16.982 ± 3.049 %
`intuitive-accuracy`	45.780 ± 5.598 %

Appendix F shows a figure where we investigate whether selecting a smaller number of LDA components may lead to better scores. Selecting the two first components yields a

`balanced-accuracy` of $27.165 \pm 6.495\%$ and an `intuitive-accuracy` of $54.193 \pm 7.301\%$.

These results are however not sufficient to ensure patient safety. The third option will therefore be considered.

Applying an LDA transformation for dimensionality reduction projects data into a K-1 feature subspace, where K is the number of classes. As this work includes 6 different classes (if we consider ramp ascent), the data is projected into a 5-dimension subspace. Therefore no feature selection was applied after transforming the data with an LDA. However, applying PCA alone may project data into a larger dimensional subspace, depending on the percentage of variance that is preserved. Assessing the number of dimensions that should be kept can be done through the computation of the cumulative explained variance ratio. It is the cumulative sum of the variance explained by each component. From this information, shown in Figure 4.6, one can identify the smallest number of components that is needed to recover a certain percentage of the dataset variance.

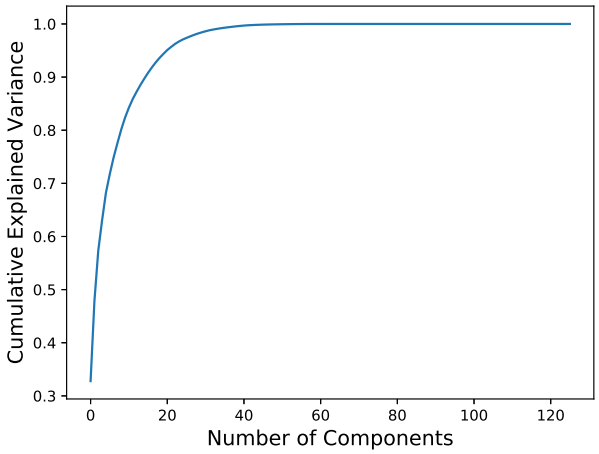


Figure 4.6: Explained variance of the training dataset

Figure 4.6 shows that approximately 50 components are sufficient to recover almost 100% of the dataset variance. More precisely, 47 components contain 99.9% of the variance.

When working with that many features, a feature selection may be relevant. Indeed, some of the 47 features may be redundant and they may alter the performance of the classifier. Also, some combinations of the features may summarize most of the information while using a smaller amount of components may bring down the computational time. For these reasons, feature selection may be one of the most important steps in classification and should not be underestimated. Results of the feature selection methods presented in Chapter 3 are shown in Table 4.3.

It should be noted that features selected through the wrappers method were selected by ranking features based on their performance score on 3 circuits: the circuits performed by subject 1 at normal and fast gait cadences and subject 2 at slow gait cadence. The scores observable in Table 4.3 are thus averaged over the 3 remaining testing sets. On the other hand, as they require no circuit sets to perform feature selection, scores for correlation, mutual information and forward greedy mutual information were averaged on the 6 testing sets. It should be mentioned that, in the following paragraphs, the wrapper

Table 4.3: Results of the different feature selection method after applying PCA

Feature Selection Method	**Selected Features**	**Scoring function**	**Score (%)**
Correlation	(6, 4, 5, 0, 30, 1)	`balanced-accuracy`	22.961 ± 6.067
		`intuitive-accuracy`	46.104 ± 8.274
MI	(4, 0, 30, 2, 1, 21)	`balanced-accuracy`	19.381 ± 6.449
		`intuitive-accuracy`	34.096 ± 11.138
Forward Greedy MI	(18, 10, 7, 25, 1, 2)	`balanced-accuracy`	24.077 ± 9.963
		`intuitive-accuracy`	28.679 ± 11.529
Wrapper with balanced score as relevance criterion (Wrapper 1)	(0, 29, 31, 18, 40, 11)	`balanced-accuracy`	20.886 ± 3.993
		`intuitive-accuracy`	24.117 ± 6.663
Wrapper with intuitive score as relevance criterion (Wrapper 2)	(6, 4, 0, 34, 29, 18)	`balanced-accuracy`	24.754 ± 1.158
		`intuitive-accuracy`	41.457 ± 5.205

approach with the balanced score as relevance criterion will be named wrapper 1 and the wrapper approach with the intuitive score as relevance criterion will be named wrapper 2.

Table 4.3 shows the accuracy scores if one selects 6 features with each method. This number of features was set as threshold to avoid working with a conditional probability table of more than 10^6 rows ($\texttt{n\_categories}^{\texttt{n\_features}}$) in the DBN. However, an example of working with the 7 features selected by wrapper 2 is shown in Appendix G. This did not give substantially better results. It is possible, though, that among the 6 selected features, some may be redundant for inference on the test sets. Selecting a lower number of features may, therefore, yield better results.

To take possible redundancies into account, the scores from Figure 4.7 were computed by iteratively adding one of the 6 selected features (see Table 4.3) for each feature selection method. Indeed, it is possible that particular combinations of features yield better results. A forward greedy search was, therefore, carried out so that at each iteration, the algorithm added the feature that, combined with the others, demonstrated the best results both in terms of intuitive and balanced score. In Figure 4.7, this "best" feature is iteratively added. Besides, in Table 4.3, the selected features are presented in the order in which they have been added by the forward search. Classification scores must be evaluated in light of the level-ground walking classification ratio. This corresponds to the number of predicted level-ground walking tasks divided by the total number of predicted tasks. Indeed, although walking is the most accomplished task among those investigated, a strategy that simply classifies every step as level-ground walking should not be selected. As seen in Figure 4.7, we reach a good trade-off by selecting the 3 features iteratively added among those selected by wrapper 2 (violet curves). As a matter of fact, it yields a balanced-accuracy score of $32.398 \pm 8.986\%$ and an intuitive-accuracy score of $48.764 \pm 5.831\%$, all the while not overestimating the number of detected level-ground walking tasks. The standard deviations associated to these scores are, however, relatively high meaning that the results may considerably vary between different testing circuits. Although these numbers are not sufficient to ensure patient safety, they are the best results obtained if a weighted average between the `balanced-score` and the `intuitive-score` is chosen as a selection criterion, while also considering the level-ground walking ratio. This strategy will therefore be pursued for the following steps of this work. The features that will be selected are (6, 4, 0).

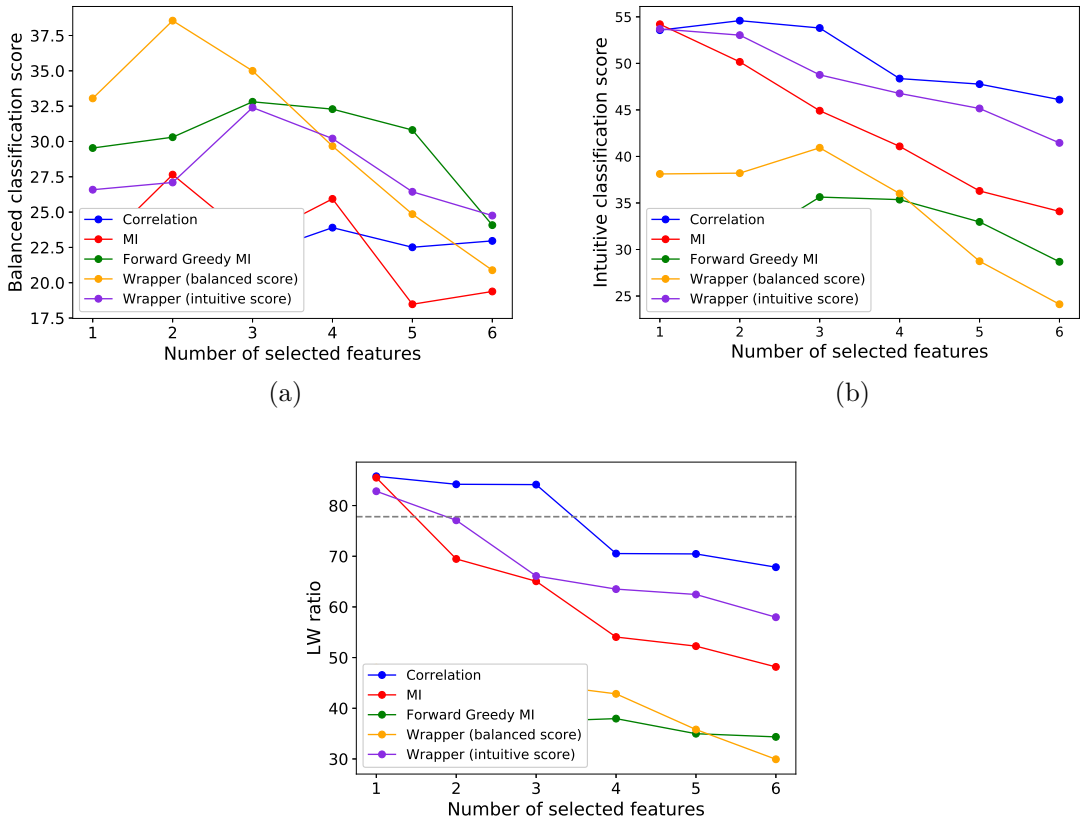


Figure 4.7: Classification scores and ratio between the number of predicted level-ground walking tasks and the total number of predicted tasks. The scores and the level-ground walking ratio were evaluated while keeping different numbers of features among those selected with the feature selection methods from Table 4.3. In (a), the balanced score is evaluated, in (b) the intuitive score is evaluated and (c) shows the ratio between the number of detected level-ground tasks and the total number of detected tasks. The dashed gray line in (c) indicates the true level-ground walking ratio, averaged over all circuits.

Along with the classification scores, it is also important to evaluate which tasks the algorithm fails to discriminate as well as the tasks that are confused. The confusion matrix, shown in Figure 4.8, brings that information. The elements in this confusion matrix C are defined such that $C_{i,j}$ corresponds to the ratio of the observations known to be in group i and predicted to be in group j [52].

By observing this confusion matrix, it appears that the algorithm generally succeeds in classifying level-ground walking tasks. However, the other tasks are also generally classified as level-ground walking. One can notice that the algorithm also tends to classify other tasks as stair ascent tasks. The diagonal of this matrix is apparent, meaning that tasks are sometimes correctly classified. However, these percentages need to be strongly reinforced. It should also be noted that standing mode is never identified. As a reminder, some data was lost during the experiments due to a misconnection between the IMUs and the program that collected the data. This explains why the sum of $C_{i,j}$ elements, for i corresponding

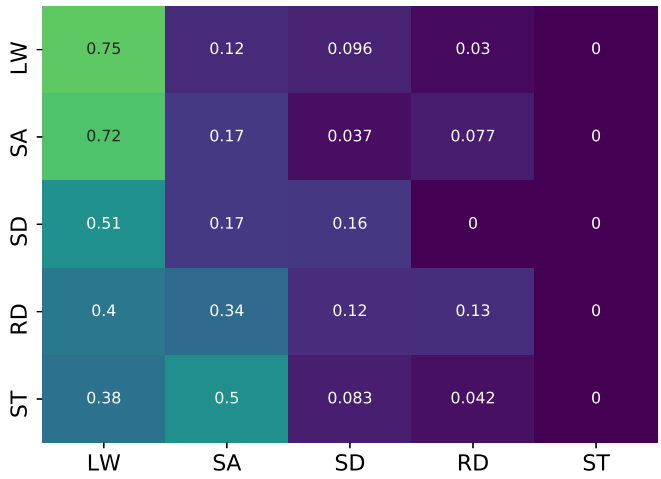


Figure 4.8: Confusion matrix averaged over the 3 testing datasets while the model was built based on the 3 features selected by the forward search on the features selected by wrapper 2.

to the stair descent mode, is not equal to 1. Indeed, while subject 1 was performing the circuit at slow gait cadence, the connection was lost before she could perform the stair descent task.

4.3 Influence of the delay

In [24], the authors observed a significant increase in the performance of the algorithm by delaying the algorithm decision by 90 [ms]. This delay results in a time analysis window that spans the gait event and the features are, therefore, computed by taking into account changes occurring at gait events. This strategy has also been implemented in this work. This was achieved by making the algorithm wait for 90 [ms] after detecting an event, before computing the features and performing inference.

Table 4.4: Results for the delayed approach when applying PCA+LDA and when applying the different selection methods after PCA

Feature Selection Method	**Selected Features**	**Scoring Function**	**Score (%)**
PCA + LDA	(3, 0, 4, 1, 2)	`balanced-accuracy`	$18.711 \pm 6.622\%$
		`intuitive-accuracy`	42.403 ± 7.850
Correlation	(4, 5, 6, 30, 1, 0)	`balanced-accuracy`	20.493 ± 6.421
		`intuitive-accuracy`	40.842 ± 12.822
MI	(4, 2, 30, 21, 0, 1)	`balanced-accuracy`	27.268 ± 9.050
		`intuitive-accuracy`	36.862 ± 11.492
Forward Greedy MI	(2, 10, 7, 1, 25, 18)	`balanced-accuracy`	18.995 ± 7.856
		`intuitive-accuracy`	31.714 ± 10.735
Wrapper with balanced score as relevance criterion (Wrapper 1)	(22, 16, 27, 8, 12, 35)	`balanced-accuracy`	32.294 ± 15.769
		`intuitive-accuracy`	30.433 ± 9.233
Wrapper with intuitive score as relevance criterion (Wrapper 2)	(2, 21, 18, 46, 1, 45)	`balanced-accuracy`	16.259 ± 1.207
		`intuitive-accuracy`	25.819 ± 7.314

This delayed approach yields better results for some methods such as the wrapper based on the balanced score (wrapper 1) or the MI. However, for other methods, it yields lower

results such as for the correlation or for wrapper 2. In general, we do not observe a net increase in performances. These results do not match those given in the conclusion from [24]. As previously, it is though important to evaluate these numbers in regards of the level-ground walking classification ratio. In addition, the results computed when we consider using a smaller number of features are shown in Figure 4.9.

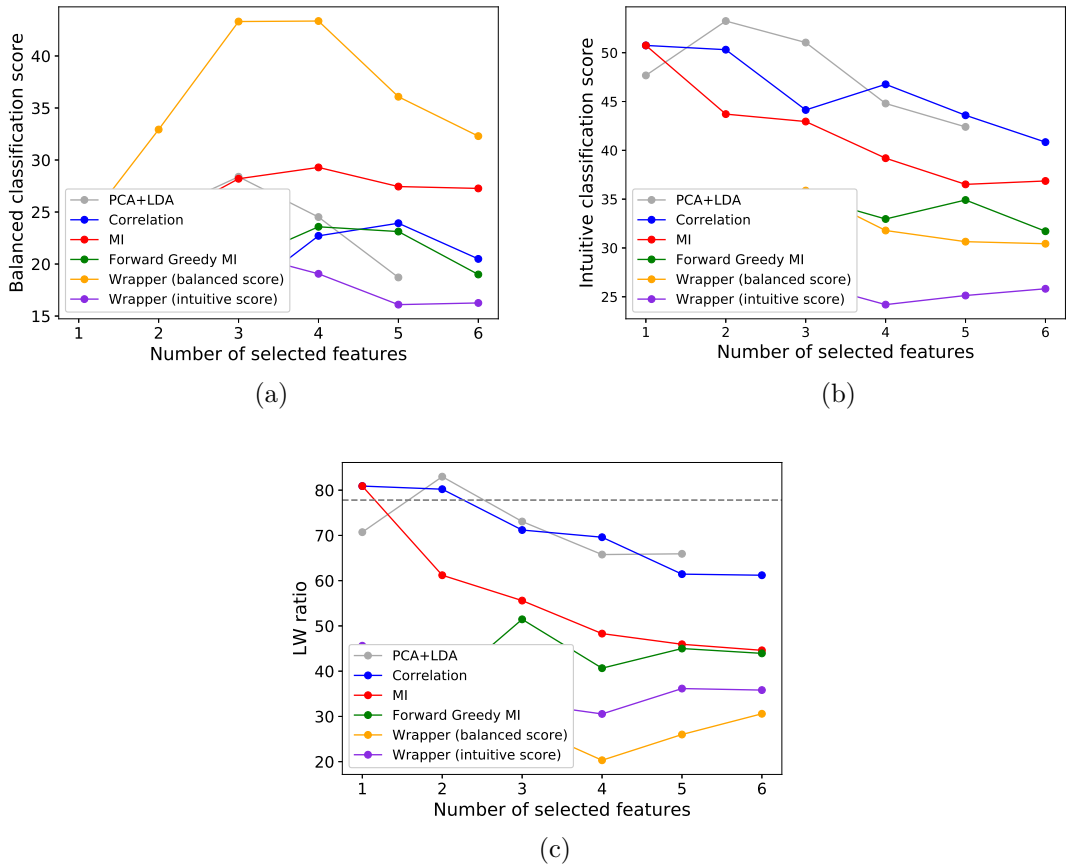


Figure 4.9: Classification scores and ratio between the number of predicted level-ground walking tasks and the total number of predicted tasks. The scores and the level-ground walking ratio were evaluated for the delayed approach while keeping different numbers of features among those selected with the feature selection methods and with the PCA+LDA approach from Table 4.4. In (a), the balanced score is evaluated, in (b) the intuitive score is evaluated and (c) shows the ratio between the number of detected level-ground tasks and the total number of detected tasks. The dashed gray line in (c) indicates the true level-ground walking ratio, averaged over all circuits.

Figure 4.9 shows that a model based on the 3 features, which were selected by the PCA+LDA approach, yields a good trade-off (gray curves) but tends to underestimate the number of level-ground walking tasks. Indeed, this approach reaches a balanced accuracy score of $28.389 \pm 11.057\%$ and an intuitive accuracy of $51.043 \pm 8.710\%$. However, given the high standard deviations associated with these scores, it may be preferred to use only the 2 first features selected by the PCA+LDA approach. This selection reaches a balanced score of $24.839 \pm 5.494\%$ and an intuitive score of $53.225 \pm 6.120\%$ while overestimating the number of level-ground walking tasks by $\sim 5\%$.

4.4 Computational time

The computational time that was needed to build the model and to infer a new task was evaluated. The DBN builds a conditional probability table with more or less columns, depending on the number of features and on the number of categories. In this work, the number of categories per feature was set at 10 while the number of selected features could change. Figure 4.10 illustrates the time (in seconds) that was necessary to build the model. As it can be seen, this time grows exponentially with the number of features.

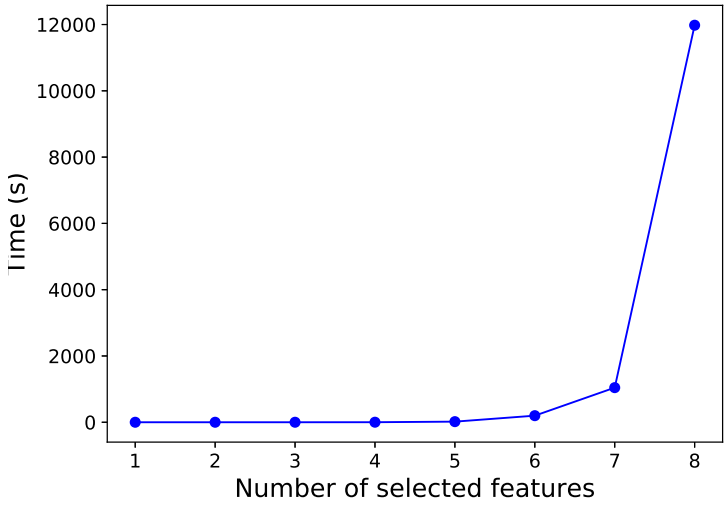


Figure 4.10: Computational time that was necessary to build the model while working with an increasing number of features.

The computational time for inferring the new task given the observations of the features was also measured. This matter is important for the future applications of this algorithm as it could be used for the control of an active prosthesis when the user is deambulating. The computational cost was evaluated over all inferences and in all the circuits. The model was built based on the 3 features selected among those selected by wrapper 2, in the non-delayed approach. The average computational time obtained was of 0.00145 [s] per inference.

Chapter 5

Discussion

This chapter first proposes a discussion that aims at explaining the results obtained in the previous chapter. It also shows that DBNs are promising algorithms for locomotion intention detection in real-time. Afterwards, the limitations related to this work and room for improvement are presented.

5.1 Performances of the algorithm

The previous chapter showed that the implemented algorithm did not perform as well as expected from the literature. Some of the best results were obtained by applying a PCA and selecting 3 features among those selected by the wrapper 2 approach (wrapper with the intuitive score as relevance criterion). These 3 features were selected by performing a forward search that iteratively added features maximizing both the `balanced-accuracy` and `intuitive-accuracy`. In the literature, however, the authors compare the performance of their algorithm in terms of overall accuracy, transitional accuracy and steady-state accuracy. As a reminder, the steady-state accuracy is the ratio between the number of correctly classified steady-state steps (steps not occurring at a transition between 2 locomotion modes) and the total number of steady-state steps [18]. The transitional accuracy is the ratio between the number of correctly classified transitional steps and the total number of transitional steps. The overall accuracy is defined as a combination of steady-state accuracy and transitional accuracy [15]. Given Equation 2.1, we can state that it corresponds to the ratio between the number of correctly classified steps (either steady-state or transitional steps) and the total number of steps. This accuracy measure does not involve weighted classes as for the `intuitive-accuracy` and `balanced-accuracy`. In order to compare our results with those presented in the reviewed studies, the steady-state accuracy, the transitional accuracy and the overall accuracy were computed after applying a PCA and selecting 3 features among those selected by the wrapper 2. These accuracy measures are given in the following table.

Table 5.1: Overall, steady-state and transitional accuracy values obtained after applying a PCA and selecting 3 components among those selected by wrapper 2.

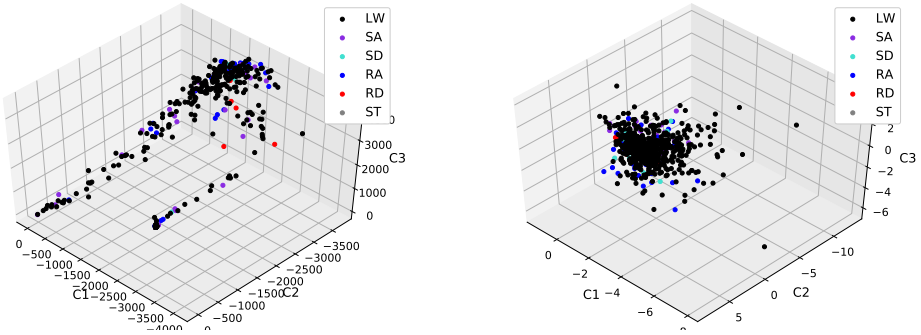
`overall-accuracy`	$62.477 \pm 11.078\%$
`steady-state-accuracy`	$64.699 \pm 10.427\%$
`transitional-accuracy`	$35.582 \pm 18.581\%$

These results are indeed lower than those obtained in [24] where the authors managed to

reach an overall accuracy of 99.02%, a steady-state accuracy of 99.22% and a transitional accuracy of 99.35%. An assumption, explaining the differences between the results from this work and those from [24], will be exposed in the following paragraphs.

Given the poor classification results obtained, the hypothesis that the training data was not representative of the testing data became apparent. This assumption translates into a difference between the distribution of the training set and of the testing sets and is supported by the subsequent thought process.

We have observed that the application of a PCA before an LDA for dimensionality reduction yielded better results. This finding does not seem consistent with Figures 4.3b and 4.5a, since Figure 4.3b, which represents the training data after applying an LDA transformation alone, shows a better class separability. This can possibly be explained by the fact that the training set and the testing sets are too differently distributed. If so, applying an LDA alone will overfit the training data and will therefore yield poor results on the testing data. Applying a PCA beforehand will reduce the dimensionality before computing the components of the LDA and probably reduce this possible overfitting. This reasoning is supported by Figures 5.1a and 5.1b. From these figures, it appears that applying an LDA alone transforms samples in such a way that the values attributed to those samples in each component become extreme compared to what was obtained for the training samples (see Figure 4.3b). This shows that the LDA transformation does not have the same effect on the testing data. Such extreme values are not observed for the successive application of PCA and LDA (Figure 5.1b).



(a) (b)

Figure 5.1: Testing data along the 3 first components obtained through LDA (a) and through successive PCA and LDA (b).

In order to verify that the difference in distribution between the training set and the testing sets is at the origin of the poor classification results, a k-fold cross-validation was performed on the training set. It consists in dividing the training set into k subsets and then selecting k-1 subsets to train the model. The remaining one is used as the validation set. This means that the model is built on the k-1 subsets and tested on the last one. This action is repeated until each subset has been selected once as the validation subset. In this work, a 10-fold cross-validation was carried out. First, the cross-validation was applied on data transformed by a LDA alone. Given 4.3b, if the training sets obtained with the 10-fold cross-validation are representative of the validation sets, this approach should reach high classification results. These results are shown in Table 5.2.

The scores obtained here are substantially better than when applying the LDA transformation on the training set and then performing inference on the testing sets. This supports

Table 5.2: Results of the 10-fold cross-validation while applying a LDA alone.

`balanced-score`	$88.058 \pm 1.958\%$
`intuitive-score`	$85.761 \pm 1.949\%$
`overall-score`	$86.514 \pm 1.795\%$

our previous assumption that those two sets are distributed differently. Besides, those results are consistent with Figure 4.3b. Therefore, given 4.5a, a 10-fold cross-validation carried out on successive PCA and LDA should reach lower results than in Table 5.2. This is indeed observable in Table 5.3.

Table 5.3: Results of the 10-fold cross-validation while applying a PCA+LDA.

`balanced-score`	$66.889 \pm 3.400\%$
`intuitive-score`	$62.236 \pm 4.406\%$
`overall-score`	$62.997 \pm 3.607\%$

5.2 Influence of the delay

From [24], it was expected that introducing a delay would yield substantially better results. Such results were however not observed in this study. This could be again explained by the fact that the training data is not representative of the testing data. Indeed, a clear separation between classes was not observed in the testing sets, even after applying successive PCA and LDA. This appears in Figure 5.2. Therefore, the benefits of a timing window spanning the gait event cannot be observed.

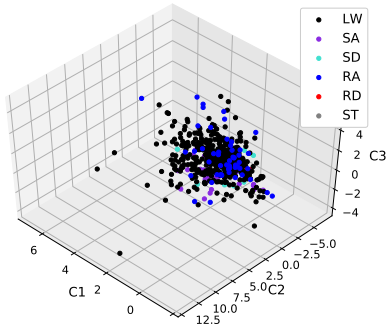


Figure 5.2: Testing data for the delayed approach along the 3 first components selected by the forward search. The components were obtained through successive PCA and LDA.

5.3 Real-time classification decisions

In many of the reviewed articles [3, 19, 23, 24], the importance of implementing a real-time classifier was highlighted. In order to obtain smooth and safe transitions between prosthesis modes when the lower-limb amputee is deambulating, the algorithm must be able to take decisions almost in real-time. The algorithm that was implemented fulfills well this important specification. Indeed, as mentioned in the previous chapter, it takes approximately 0.00145 [s] for the DBN to infer a new task from the recorded features. In [3] and in [24], the authors voluntarily introduced delays of 90 [ms] and 125 [ms] respectively. Both articles support that introducing such kinds of delays would give enough time to adjust the parameters of the prosthesis for the next locomotion mode. Indeed, this adjustment could occur before a critical timing after which the walking balance of the

user would be impaired [24]. As our algorithm only introduces a delay of approximately 0.00145 [s], we can state, based on the conclusion of these articles, that it can enable safe and seamless transitions.

5.4 Study limitations

The main limitations of this study are linked to the small number of experiments that were conducted. As a reminder, only a limited number of subjects were able to participate in the experiments due to the particular COVID-19 situation and the requirement to use a specific instrumented insole. Indeed, only subjects with an approximate foot size of 40 were allowed to participate. The literature often refers to experiments conducted on 6 [6] or even 8 subjects [23]. In these experiments, the subjects performed many more trials than for this Master Thesis. The primary objective of this thesis was to implement a general, yet user-dependent, method that could be used, with some adaptations, for the control of an active prosthesis. By collecting more training and testing data, it would have been possible to obtain a training set more representative of the testing set. However, given the amount of data collected, such a generalization was not feasible.

Besides, if it had been possible to collect more data, mode-specific DBNs able to produce higher classification results could have been trained and used for inference. [23, 24, 25] have indeed shown this approach to be efficient. It was also demonstrated in [18] that building models based on training data collected from multiple timing windows sliding along the training signals yields lower classification scores than training the model based on two single timing windows captured at heel contact and at toe-off. This approach was also not implemented. Indeed the number of gait events detected in the training signals was insufficient to build an acceptable training set.

Detecting gait events constituted one of the main challenges of this work. In Chapter 3, we saw that the gait phase estimator struggled to estimate correctly the phase of the signal. Therefore, misdetections of the toe-off and of the mid-swing sometimes arose despite an important work of fine-tuning. This issue needs to be further addressed in studies on the same topic. However, since the general approach was chosen for the implementation of DBNs, the impact of these errors is limited to the inference on the testing data. The conditional probability table of the DBN does not reflect these errors.

In addition, we first assumed the conditional independence between the features given the task, in order to overcome the number of samples necessary to fill the high dimensional space of features. Though strong, this hypothesis is central to working with Bayesian Networks. Then, we made the assumption that the transition probability matrix and marginal probabilities have been only inspired from data and based mainly on intuition, personal knowledge and personal observations. These probabilities can nevertheless easily be adapted to other applications. Besides, before building the conditional probability table of the DBN, the data must be discretized, which results in a loss of information. In this work, the number of categories was set at 10 per feature. In further studies, an implementation allowing to work with more than 10 categories per feature may need to be considered. However, Figure 4.4 showed that a good separability between classes could be maintained, despite the categorization. One may also consider using other types of models that do not require a previous discretization, such as SVM, KNN, and so forth.

In this study, only a fixed analysis window size of 300 [ms] and a fixed window increment were used, based on findings from [24]. However, others studies such as [18] used these two parameters as search parameters to improve the performance of their algorithm. This

could be the topic of a subsequent work for purposes of enhancing the results obtained in this Master Thesis. A future work could also be dedicated to the choice of features that should be computed.

Moreover, given the simple structure of the Dynamic Bayesian Network that was implemented, no python library was used for its implementation. It should however be adapted to deal with more complicated structure.

5.5 Perspectives

The results that were obtained by performing a 10-fold cross-validation on the LDA-transformed data are promising. These results show that Dynamic Bayesian Networks have the capacity to discriminate a subject's locomotion intentions to a high degree and to perform almost real-time inference. By conducting further experiments for collecting both training and testing data, on a wider population of subjects, Dynamic Bayesian Networks could have the potential for clinical applications.

Furthermore, as suggested by several authors, other types of sensors could be used. The instrumentation that was used in this work had the benefit of being light and easy to handle. As a matter of fact, only three IMUs and one instrumented insole were necessary. However, joint angles, torques, motor currents, interaction forces, and so on could be recorded and bring relevant information [18, 19, 23]. Most of these signals are only measurable from a prosthesis and therefore could only be recorded in a study involving amputee subjects. However, these possibilities should be kept in mind. Angles could be measured on healthy subjects by using goniometers but these sensors could be rather expensive. They could also be computed by using the method suggested by S. Slajpah et al. [53] that uses IMUs measurements and a Kalman filter algorithm for estimating the orientation of leg segments. These orientations would allow to retrieve the angles between segments, and thus the angles of joints. Moreover, EMG data combined with mechanical data could also be used as suggested by [15, 22, 27]. Targeted muscle re-innervation [27] could be considered for amputated subjects for whom the recovery of impaired muscle signals is not possible.

Depth sensing, as performed in [54], could also be integrated. This would imply integrating information about the environment. Since gait patterns generally vary between different subjects, days or even trials, it is necessary to integrate information that does not suffer from this variability. Environment information offers the benefit of being subject independent and its integration could induce potential improvements [55]. The authors from [54] used a Microsoft Kinect to record depth data and performed image segmentation for depth information extraction. Their strategy succeeded in recognizing a stair approach with an accuracy of 98.8%.

Finally, in order to enlarge the methods implemented in this work for clinical applications, a user-independent strategy should be investigated. Indeed, as suggested above, a high number of training experiments need to be conducted in order to obtain training sets that are representative of testing sets. It would be inconvenient to conduct such training experiments for each new user, as they would involve several hours of in-lab walking. [17] and [25] investigated user-independent strategies where data from a pool of subjects was used to train an algorithm, which was tested on a new user. These studies demonstrated that, by introducing only little training data from the new user, an overall accuracy of 86% could be obtained [17]. Such an approach would however require working with an important population of subjects to collect the training data.

Conclusion

The aim of this work was to develop an algorithm, based on available literature, which could prove itself efficient in detecting the locomotion intentions of a subject. The tasks that were investigated involved elements typically encountered in daily life activities. These are level-ground walking, standing, stair ascent and descent, and ramp ascent and descent. The algorithm that was developed was chosen after much consideration, by comparing different studies and highlighting their main characteristics and results. One algorithm that stood out in the literature was the Dynamic Bayesian Network. These networks allow to infer current values by integrating past and current information, and have been used in the context of numerous studies regarding locomotion intention recognition [15, 22, 23, 24]. One particular study, from [24], exhibited remarkably low classification errors when using Dynamic Bayesian Networks for intent recognition. In this study, however, ramp ascent steps were often classified as level-ground walking steps. Since these two modes show similar impedance parameter settings [24], this was not viewed as a problem for user deambulation. Besides, the authors observed that performances increased when using information from signals spanning gait events in order to discriminate among locomotion tasks. This was achieved by introducing time analysis windows that overlapped the gait events, and thus by introducing a short delay in the classifier decisions.

Despite this delay, the authors reported that most transitions, from one locomotion mode to another, occurred smoothly. It is actually a point of major importance that the classifier decisions are taken almost in real-time. This avoids impairing the user's walking balance and thereby reduces the risk of falling. This is where the crux of the matter lies when it comes to detecting the locomotion intention of lower-limb amputees. To the best of our knowledge, there is currently no consensus as to the best method to detect locomotion intentions. Nevertheless, as the number of lower-limb amputees continues to rise [3], defining a complete method for online detection of locomotion intentions is of major importance. Such a method could benefit numerous patients who have lost a limb due to cancer, vascular disease, malformation, and so forth, allowing them to regain mobility for daily life activities and to participate in social life.

Our work was carried out with this in mind. The experiments were conducted on two able-bodied subjects, wearing only a few sensors, including 3 IMUs and an instrumented insole. This light instrumentation represents a key point for the development of a commercially available strategy. Though this aspect was not the goal of this work, it would prove important for elaborating a practical and affordable strategy. The results exposed in Chapter 4 and discussed in Chapter 5, have demonstrated that the method investigated here needs further experimentation, on a larger number of subjects, in order to be validated. However, promising results, explained in Chapter 5, do pave the way for further investigation of locomotion intention detection through the use of Dynamic Bayesian Net-

works and light instrumentation. Moreover, this work has demonstrated that it is possible to implement such a method almost in real-time.

In Chapter 5 several study limitations and perspectives were highlighted. We could, for example, emphasize the need to investigate the optimal window size and window increment from which features are extracted. Such research was not part of this work but fine-tuning these elements would probably enable the classifier to reach higher classification scores. Moreover, sensor fusion also has the potential to improve results in intention detection. Further potential explorations include neuromechanical fusion [3, 17, 55], as well as environment sensing, which could be combined with neural and mechanical information [54, 55]. This would however very likely entail a heavier instrumentation.

To conclude, we can state that this work demonstrates the potential in using Dynamic Bayesian Networks to address the issue of recognizing human locomotion intentions in real-time. Given our particular work, such an algorithm could possibly succeed in recognizing locomotion tasks while using only a small number of mechanical sensors. Subsequent studies, however, involving further experiments on a greater number of subjects, will be necessary to validate this conclusion.

Bibliography

[1] Rohit Gupta and Ravinder Agarwal. Continuous human locomotion identification for lower limb prosthesis control. *CSI Transactions on ICT*, 6(1):17–31, March 2018.

[2] P. W. Moxey, P. Gogalniceanu, R. J. Hinchliffe, I. M. Loftus, K. J. Jones, M. M. Thompson, and P. J. Holt. Lower extremity amputations - a review of global variability in incidence: Lower extremity amputations-a global review. *Diabetic Medicine*, 28(10):1144–1153, October 2011.

[3] D. C. Tkach and L. J. Hargrove. Neuromechanical sensor fusion yields highest accuracies in predicting ambulation mode transitions for trans-tibial amputees. In *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 3074–3077, Osaka, July 2013. IEEE.

[4] F. Sup, H.A. Varol, J. Mitchell, T.J. Withrow, and M. Goldfarb. Preliminary Evaluations of a Self-Contained Anthropomorphic Transfemoral Prosthesis. *IEEE/ASME Transactions on Mechatronics*, 14(6):667–676, December 2009.

[5] Michael R Tucker, Jeremy Olivier, Anna Pagel, Hannes Bleuler, Mohamed Bouri, Olivier Lambercy, José del R Millán, Robert Riener, Heike Vallery, and Roger Gassert. Control strategies for active lower extremity prosthetics and orthotics: a review. *Journal of NeuroEngineering and Rehabilitation*, 12(1):1, 2015.

[6] Aaron J. Young, Ann M. Simon, Nicholas P. Fey, and Levi J. Hargrove. Intent Recognition in a Powered Lower Limb Prosthesis Using Time History Information. *Annals of Biomedical Engineering*, 42(3):631–641, March 2014.

[7] Purves D. et al. *Neurosciences*. Neurosciences and cognition. de boeck, 4th edition, 2012.

[8] Gait, March 2015. Library Catalog: clinicalgate.com.

[9] R. Baker. Temporal spatial data, the gait cycle and gait graphs.

[10] Fundamentals of Human Gait, December 2016.

[11] Sharon R. Perry-Rana, Terry J. Housh, Glen O. Johnson, Anthony J. Bull, Joseph M. Berning, and Joel T. Cramer. MMG and EMG responses during fatiguing isokinetic muscle contractions at different velocities. *Muscle & Nerve*, 26(3):367–373, September 2002.

[12] YANG Xiaozhou. Linear Discriminant Analysis, Explained, May 2020. Library Catalog: towardsdatascience.com.

[13] Michał Oleszak. Linear Classifiers: An Overview, February 2020. Library Catalog: towardsdatascience.com.

[14] Michel Verleysen. ELEC2870 - Machine learning: regression and dimensionality reduction. *Support Vector Machines*, page 20.

[15] Aaron J. Young, Ann Simon, and Levi J. Hargrove. An intent recognition strategy for transfemoral amputee ambulation across different locomotion modes. In *2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1587–1590, Osaka, July 2013. IEEE.

[16] He Huang, T.A. Kuiken, and R.D. Lipschutz. A Strategy for Identifying Locomotion Modes Using Surface Electromyography. *IEEE Transactions on Biomedical Engineering*, 56(1):65–73, January 2009.

[17] Aaron J. Young, Ann M. Simon, Nicholas P. Fey, and Levi J. Hargrove. Classifying the intent of novel users during human locomotion using powered lower limb prostheses. In *2013 6th International IEEE/EMBS Conference on Neural Engineering (NER)*, pages 311–314, San Diego, CA, USA, November 2013. IEEE.

[18] Aaron J. Young, Ann M. Simon, and Levi J. Hargrove. A Training Method for Locomotion Mode Prediction Using Powered Lower Limb Prostheses. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 22(3):671–677, May 2014.

[19] H.A. Varol, F. Sup, and M. Goldfarb. Multiclass Real-Time Intent Recognition of a Powered Lower Limb Prosthesis. *IEEE Transactions on Biomedical Engineering*, 57(3):542–551, March 2010.

[20] Gholamreza Khademi, Hanieh Mohammadi, Dan Simon, and Elizabeth C. Hardin. Evolutionary optimization of user intent recognition for transfemoral amputees. In *2015 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–4, Atlanta, GA, USA, October 2015. IEEE.

[21] He Huang, Fan Zhang, L. J. Hargrove, Zhi Dou, D. R. Rogers, and K. B. Englehart. Continuous Locomotion-Mode Identification for Prosthetic Legs Based on Neuromuscular–Mechanical Fusion. *IEEE Transactions on Biomedical Engineering*, 58(10):2867–2875, October 2011.

[22] A J Young, T A Kuiken, and L J Hargrove. Analysis of using EMG and mechanical sensors to enhance intent recognition in powered lower limb prostheses. *Journal of Neural Engineering*, 11(5):056021, October 2014.

[23] J A Spanias, A M Simon, S B Finucane, E J Perreault, and L J Hargrove. Online adaptive neural control of a robotic lower limb prosthesis. *Journal of Neural Engineering*, 15(1):016015, February 2018.

[24] Ann M. Simon, Kimberly A. Ingraham, John A. Spanias, Aaron J. Young, Suzanne B. Finucane, Elizabeth G. Halsne, and Levi J. Hargrove. Delaying Ambulation Mode Transition Decisions Improves Accuracy of a Flexible Control System for Powered Knee-Ankle Prosthesis. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 25(8):1164–1171, August 2017.

[25] Aaron J. Young and Levi J. Hargrove. A Classification Method for User-Independent Intent Recognition for Transfemoral Amputees Using Powered Lower Limb Prosthe-

ses. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 24(2):217–225, February 2016.

[26] Ann M. Simon, Emily A. Seyforth, and Levi J. Hargrove. Across-Day Lower Limb Pattern Recognition Performance of a Powered Knee-Ankle Prosthesis. In *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob)*, pages 242–247, Enschede, August 2018. IEEE.

[27] Levi J. Hargrove, Ann M. Simon, Aaron J. Young, Robert D. Lipschutz, Suzanne B. Finucane, Douglas G. Smith, and Todd A. Kuiken. Robotic Leg Control with EMG Decoding in an Amputee with Nerve Transfers. *New England Journal of Medicine*, 369(13):1237–1242, September 2013.

[28] Samuel Au, Max Berniker, and Hugh Herr. Powered ankle-foot prosthesis to assist level-ground and stair-descent gaits. *Neural Networks*, 21(4):654–666, May 2008.

[29] Song-Mi Lee, Sang Min Yoon, and Heeryon Cho. Human activity recognition from accelerometer data using Convolutional Neural Network. In *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 131–134, Jeju Island, South Korea, February 2017. IEEE.

[30] Yanggang Feng, Wanwen Chen, and Qining Wang. A strain gauge based locomotion mode recognition method using convolutional neural network. *Advanced Robotics*, 33(5):254–263, March 2019.

[31] K. De Koster K. Jackson M. Cnudde. Gait.

[32] L.J. Hargrove, A.J. Young, and A.M. Simon. Intuitive Control of a Powered Prosthetic Leg During Ambulation: A Randomized Clinical Trial. *Journal of Vascular Surgery*, 63(5):1405–1406, May 2016.

[33] MotionControl.com. Motor-driven ball screw.

[34] E. Quinn. Generally accepted values for normal range of motion (rom) in joints.

[35] Merck Manual. Physical therapy (pt).

[36] Luc Vandendorpe. Change of sampling rate. In *ELEC2900: Signal Processing [Course syllabus]*, pages 51–54. Université Catholique de Louvain, EPL, 2020.

[37] Benoit Macq. Digital Filters Design. In *ELEC2900: Signal Processing [Course slides]*. Université Catholique de Louvain, EPL, April 2020.

[38] John A. Lee. Unsupervised dimensionality reduction. In *LDATA2010 Information vizualization [Course Slides]*. 2018.

[39] Ashwin N. Feature Extraction Techniques: PCA, LDA and t-SNE, January 2020. Library Catalog: medium.com.

[40] Paul Brodersen. paulbrodersen/entropy_based_binning, February 2020. original-date: 2016-09-08T13:00:18Z.

[41] Michel Verleysen. Features Selection. In *ELEC2870: Machine learning, regression and dimensionality reduction [Course slides]*. Université Catholique de Louvain, EPL, 2019.

[42] D Francois, V Wertz, and M Verleysen. The permutation test for feature selection by mutual information. page 6, 2006.

[43] Edouard Couplet and Camille Draguet. Prediction of air-quality in Beijing [Project ELEC2870], December 2019.

[44] Greg Ver Steeg. gregversteeg/NPEET, April 2020. original-date: 2014-10-10T19:57:02Z.

[45] Tingfang Yan, Andrea Parri, Virginia Ruiz Garate, Marco Cempini, Renaud Ronsse, and Nicola Vitiello. An oscillator-based smooth real-time estimate of gait phase for wearable robotics. *Autonomous Robots*, 41(3):759–774, March 2017.

[46] Renaud Ronsse, Stefano Marco Maria De Rossi, Nicola Vitiello, Tommaso Lenzi, Maria Chiara Carrozza, and Auke Jan Ijspeert. Real-Time Estimate of Velocity and Acceleration of Quasi-Periodic Signals Using Adaptive Oscillators. *IEEE Transactions on Robotics*, 29(3):783–791, June 2013.

[47] Virginie Otlet. Towards artificial sensory feedback for lower-limb amputees - how does a vibrotactile stimulation of the patellar tendon influence the gait pattern of able-bodied subjects? *Ecole polytechnique de Louvain, Université Catholique de Louvain*, 2019.

[48] Devin Soni. Introduction to Bayesian Networks, July 2019. Library Catalog: towardsdatascience.com.

[49] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. Adaptive computation and machine learning. MIT Press, Cambridge, MA, 2009.

[50] Pierre Dupont. INGI2262: Bayesian Learning (II). page 30.

[51] sklearn.metrics.balanced_accuracy_score — scikit-learn 0.23.0 documentation.

[52] sklearn.metrics.confusion_matrix — scikit-learn 0.23.1 documentation.

[53] S. Šlajpah, R. Kamnik, and M. Munih. Kinematics based sensory fusion for wearable motion assessment in human walking. *Computer Methods and Programs in Biomedicine*, 116(2):131–144, September 2014.

[54] Nili Eliana Krausz, Tommaso Lenzi, and Levi J. Hargrove. Depth Sensing for Improved Control of Lower Limb Prostheses. *IEEE Transactions on Biomedical Engineering*, 62(11):2576–2587, November 2015.

[55] Domen Novak and Robert Riener. A survey of sensor fusion methods in wearable robotics. *Robotics and Autonomous Systems*, 73:155–170, November 2015.

Appendix A

Review - Pareto front

In this Appendix, one can find the scatter plot presenting some of the articles that were reviewed, positioned according to their accuracy measure and the time delay associated to their algorithm. In Figure A.1, the accuracy measure is the steady-state error and in Figure A.2, the accuracy measure is the transitional error. The article [24] is on the Pareto front in both cases.

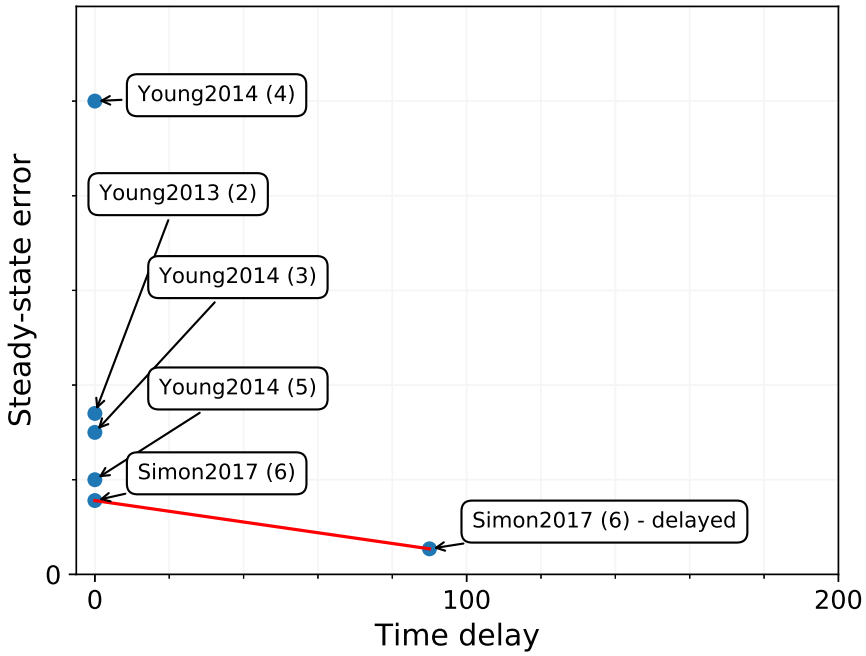


Figure A.1: Set of selected articles, positioned according to their stead-state classification error and their time delay. The Pareto front is indicated in red.

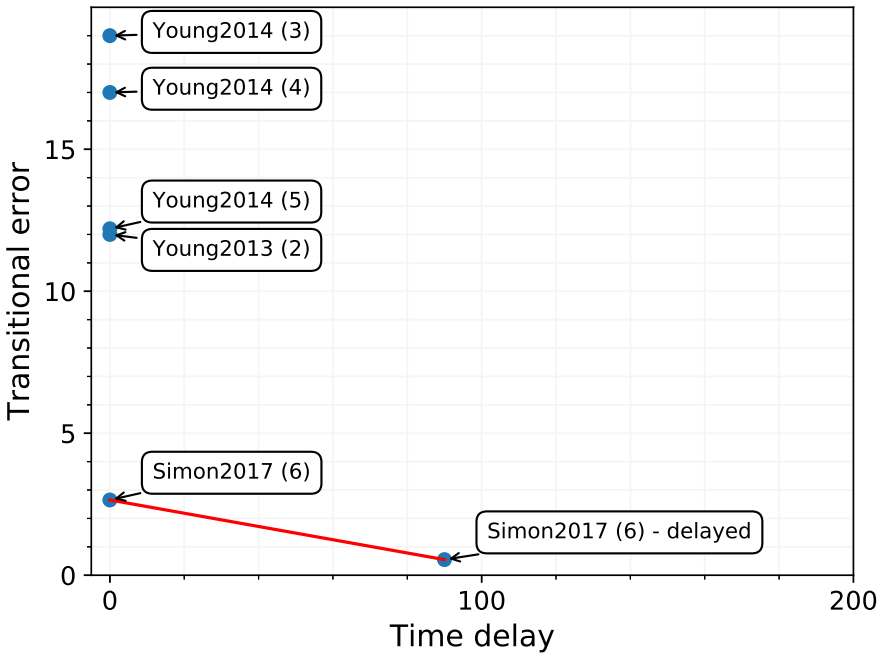


Figure A.2: Set of selected articles, positioned according to their transitional classification error and their time delay. The Pareto front is indicated in red.

Appendix B

Protocol - A complete description of the training circuit

The circuit that was performed by the subjects is entirely described here. The number of steps during level-ground walking and ramp tasks are an approximation.

- still standing
- $\sim$ 4 level-ground walking steps
- 6 stair ascent steps
- $\sim$ 17 level-ground walking steps
- $\sim$ 7 ramp descent steps
- $\sim$ 2 level-ground walking steps
- turn-over, still standing for 5 seconds
- $\sim$ 2 level-ground walking steps
- $\sim$ 7 ramp ascent steps
- $\sim$ 17 level-ground walking steps
- 6 stair descent steps
- $\sim$ 4 level-ground walking steps
- turn-over, still standing for 5 seconds
- $\sim$ 4 level-ground walking steps
- 3 stair ascent steps
- turn-over, still standing for 5 seconds
- 3 stair descent steps
- $\sim$ 4 level-ground walking steps
- still standing

Appendix C

Methods - Implementation of the operations

This Appendix describes the scripts that have been attached to this report and that are used for implementing the operations from Figure 3.4.

The script `FeaturesAcquisition` regroups a series of functions that will allow to build the training set from training signals. This includes a sliding window that takes snapshots of the signals and that computes the mean, the standard deviation, the maximum and minimum values as well as the initial and final values from each snapshot.

A second script `DBN` contains the two functions `DBNetwork` and `makeInference` that are explained in Chapter 3, in Section 3.7.2.

`FeaturesSelection` implements all the methods for feature selection that are presented in Chapter 3. This includes the correlation method, the MI method, the forward greedy MI method, the forward-backward wrappers and the forward wrappers for which the features are selected according to their position in the list of candidates.

The script `ClassificationResults` implements the dimensionality reduction and builds the general model based on the selected features. After building the model, it allows to perform inference on the testing circuits and to obtain all type of classification scores for each of these circuits.

The `CrossVal` script implements the 10-fold cross-validation. This method was used to show that the DBN has the capacity to highly discriminate among the locomotion intentions of a subject if the training data and the testing data are similarly distributed.

Finally, the `GaitPhaseEstimator` essentially contains the `adaptive_osc` function that implements an adaptive oscillator. Such an oscillator is at the core of the gait event detection block. This function has been translated from Virginie Otlet's work from Matlab to Python.

Appendix D

Methods - Gait phase learning

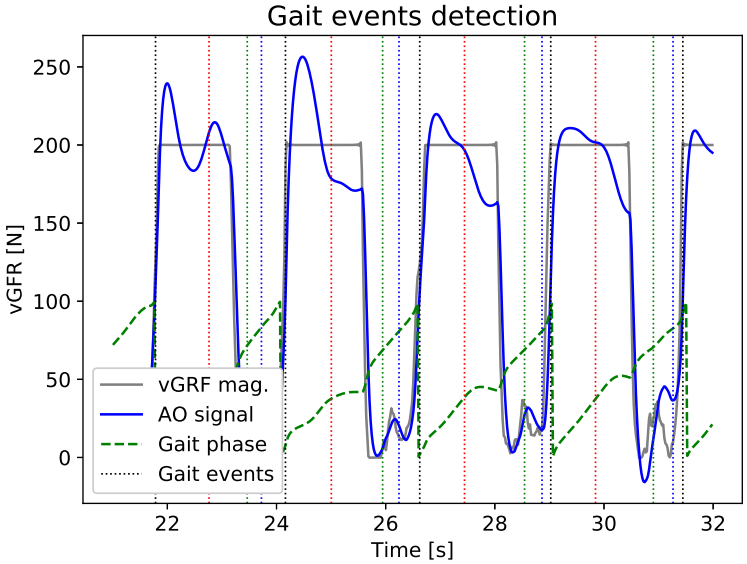


Figure D.1: Gait phase learning for subject 1 performing the circuit at a slow gait cadence. Figure (a) shows the phase learning and heel contact detection over the entire circuit and Figure (b) highlights the phase adaptation and events detection during the first steps of the circuit. Heel contacts are represented by black dashed lines, mid-stance detection by red dashed lines, toe-off by green dashed lines and mid-swing by blue dashed lines.

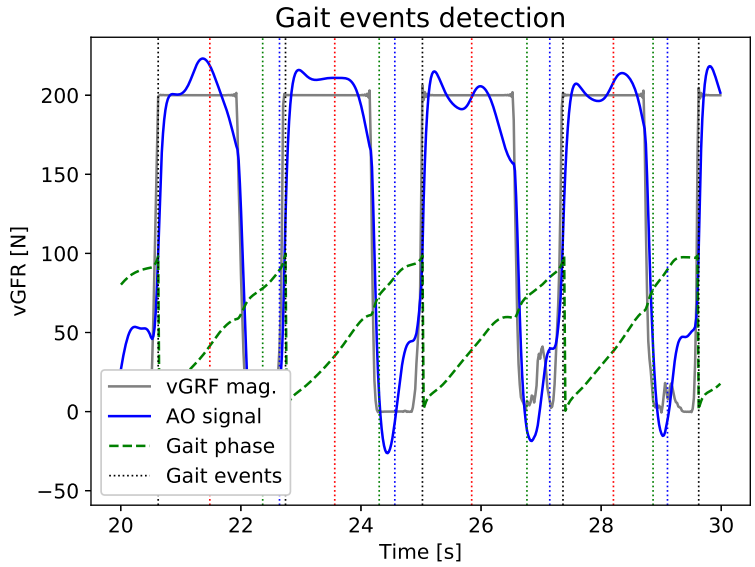


Figure D.2: Gait phase learning for subject 2 performing the circuit at a slow gait cadence.

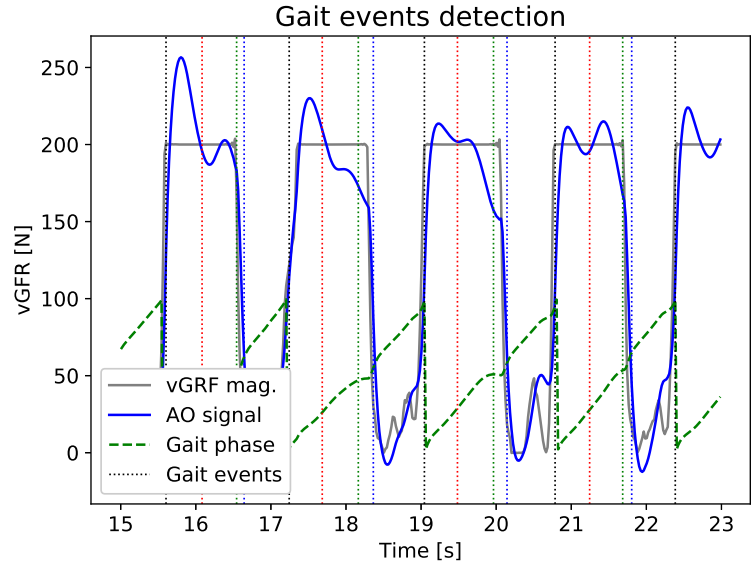


Figure D.3: Gait phase learning for subject 1 performing the circuit at a normal gait cadence.

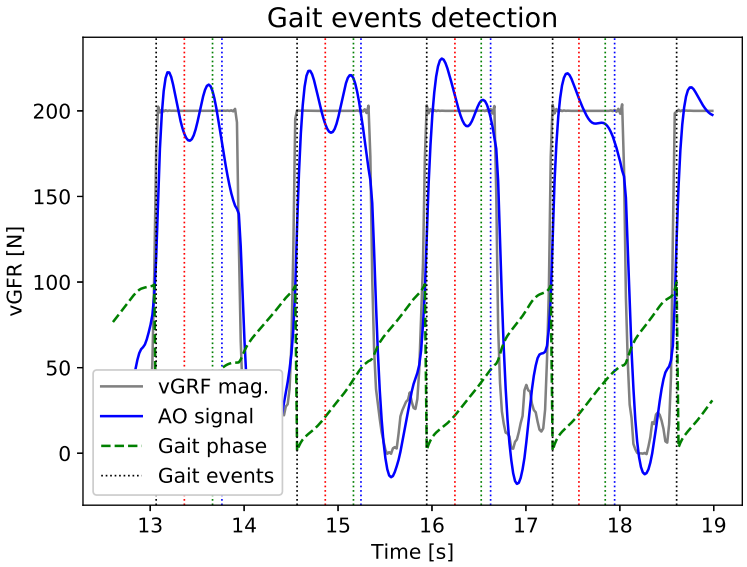


Figure D.4: Gait phase learning for subject 1 performing the circuit at a fast gait cadence.

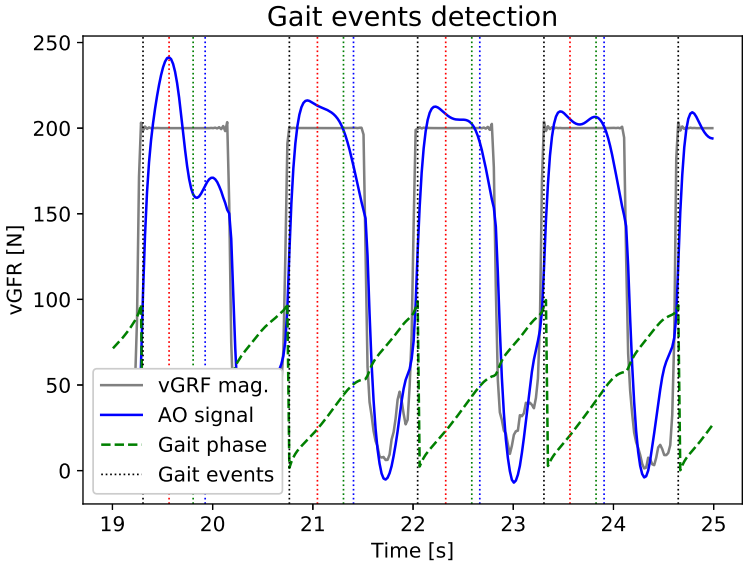
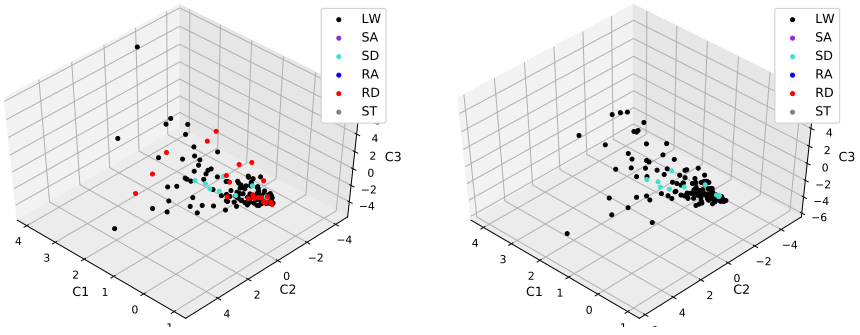


Figure D.5: Gait phase learning for subject 2 performing the circuit at a fast gait cadence.

Appendix E

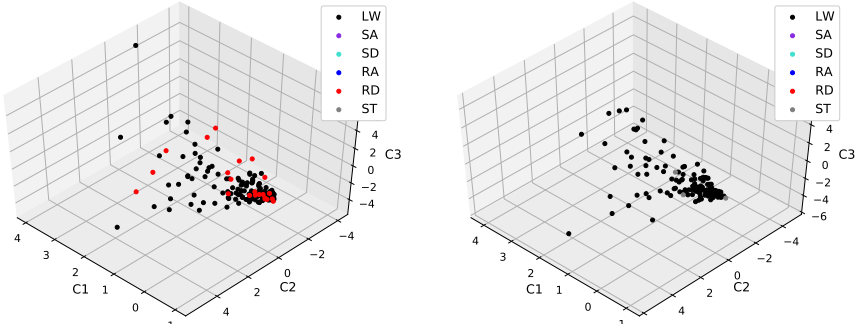
Results - Mode specific classifiers

The following figures show the training data for each of the 7 mode-specific classifiers from Table 3.4.



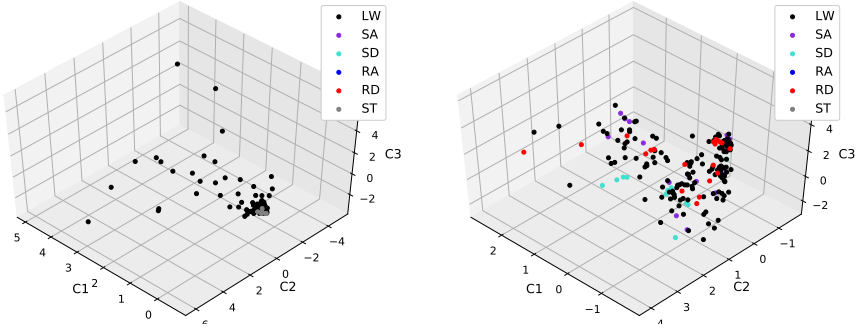
(a)

(b)



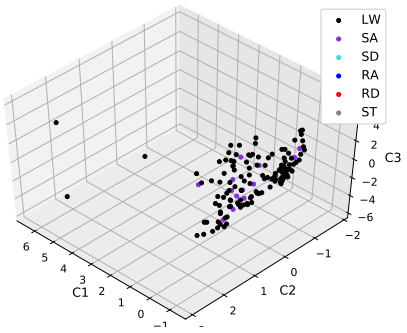
(c)

(d)



(e)

(f)



(f)

Figure E.1: Training data along the 3 first principal components. (a) represents the training data for HC_LW, (b) for HC_SD, (c) for HC_RD, (d) for HC_ST, (e) for MST_LW, (f) for TO and (g) for MSW_SA.

It can be observed from these figures that only a few training data is available for each classifier. This data has been extracted from 5 out of the 6 experiments circuits. The circuit performed by subject 1 at fast gait cadence was left for testing purposes. According to [18, 23], the features were computed from time analysis windows taken 300 [ms] before each detected gait events both for training and testing. This gives less training data than in the general approach, which was considered in this Master Thesis. For this reason, we do not expect that this training data will be very representative of the testing data.

Indeed, while performing inference on the last testing set, with selected features being (6, 4, 0), we obtained relatively poor classification results. As a reminder, those features are those selected by the forward greedy search on the wrapper approach with intuitive score as relevance criterion.

Table E.1: Classification results with general DBN after applying a PCA for dimensionality reduction and a relevant feature selection.

`balanced-accuracy`	42.575 %
`intuitive-accuracy`	65.227 %
`overall-accuracy`	71.2 %
`transitional-accuracy`	57.1143 %
`steady state-accuracy`	72.034 %
`LW ratio`	86.4 %

Those results are higher than what is obtained with the general model. However, given the number of training data and the number of remaining testing sets, this approach was considered not suited for generalization. The confusion matrix corresponding to these results is shown in Figure E.2. The great majority of tasks are classified as level-ground walking. This was expected given the training data shown in Figure E.1. The mode-specific algorithm however did recognize the standing mode task that appeared in the circuit.

Figure E.2: Confusion matrix for the mode specific classifiers, obtained by performing inference on the circuit accomplished by subject 1 at fast gait cadence.

Appendix F

Results - Forward search on the LDA components

In this Figure, the `balanced-accuracy` and `intuitive-accuracy`, while applying a successive PCA + LDA for dimensionality reduction, are presented. These results should be considered by taking into account the level-ground walking ratio, to avoid selecting an approach that reaches high results by systematically classifying tasks as level-ground walking. Indeed the level-ground walking classification ratio corresponds to the ratio between the number of detected level-ground walking tasks and the total number of detected tasks. The features were iteratively added by the forward search in the order (3, 0, 2, 4, 1).

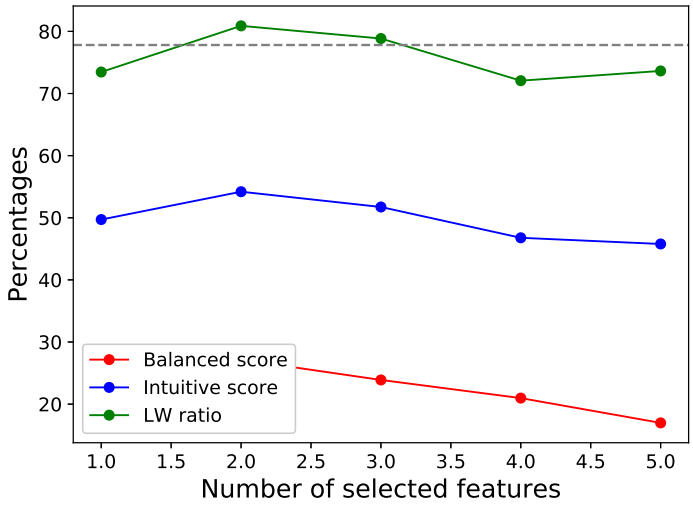


Figure F.1: Classification scores and level-ground walking classification ratio. Those were computed while keeping different numbers of components after applying a successive PCA + LDA. The dashed gray line indicates the true level-ground walking ratio, averaged over all circuits.

Appendix G

Results - Scores for 7 features

This Appendix presents the results of adding an extra-feature to the 6 selected features with the wrapper approach with intuitive score as relevance criterion. Features are selected by the forward search in the order (6, 4, 0, 5, 34, 18, 29). Results with 7 features are given in Table G.1.

Table G.1: Classification results while working with the 7 features selected by the wrapper approach with intuitive score as relevance criterion.

`balanced-accuracy`	25.090 ± 1.010 %
`intuitive-accuracy`	42.337 ± 6.303 %

Results of the iterative addition of features is shown in Figure G.1.

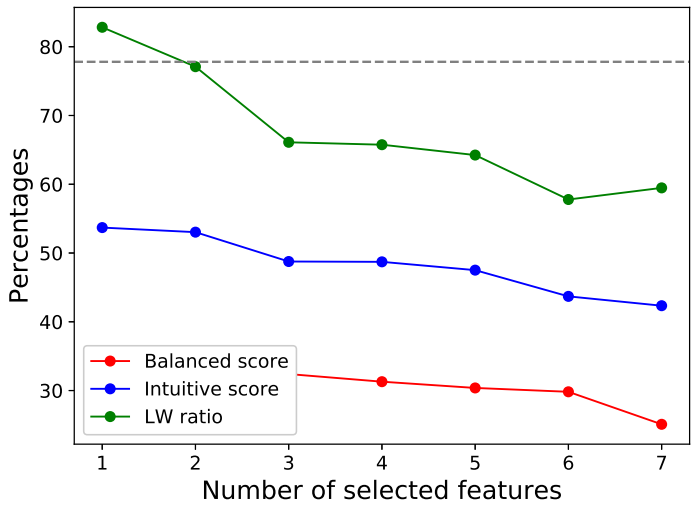


Figure G.1: `Balanced-accuracy`, `intuitive-accuracy` and level-ground walking ratio while iteratively adding features upon 7 features selected by the wrapper approach with intuitive score as relevance criterion.

UNIVERSITÉ CATHOLIQUE DE LOUVAIN
École polytechnique de Louvain
Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve, Belgique | **www.uclouvain.be/epl**