8-23-2018

An Algorithmic Framework For Gait Analysis and Gait-Based Biometric Authentication

Ioannis Papavasileiou
University of Connecticut - Storrs, ioannis.papavasileiou@uconn.edu

Follow this and additional works at: https://opencommons.uconn.edu/dissertations

Recommended Citation
https://opencommons.uconn.edu/dissertations/1925
An Algorithmic Framework For Gait Analysis and Gait-Based Biometric Authentication

Ioannis Papavasileiou, Ph.D.
University of Connecticut, 2018

ABSTRACT

Gait refers to the locomotion achieved through the movement of human limbs and is fairly unique to an individual due to the limb’s specific muscular-skeletal structure. However, conditions that affect the nervous system, such as Parkinson’s Disease (PD) and stroke, can cause significant impairment in cognition, motor skills and thus lead to gait disorders. Consequently, small or large deviations present in someone’s gait could be attributed to either their unique gait patterns or possible underlying neurological disorders. With the rapid development of wearable technologies and computing capabilities, it is now possible to quantitatively measure such deviations. In this thesis we focus on different types of deviations present in someone’s gait. We develop an algorithmic framework that identifies the deviations caused from neurological disorders, that can have applications in gait physical therapy, or from unique individual behavior, which can have applications in behavioral biometrics. In the first two parts of this thesis we present two methods for gait analysis. To objectively extract gait phases, an infinite Gaussian mixture model is proposed to classify different gait phases, and a parallel particle filter is designed to estimate and update the model parameters in real-time. To objectively classify gait disorders caused by PD and stroke diseases and to facilitate gait physical therapy, an advanced machine learning method, multi-task learning, is used to jointly train classification models of a subject’s gait. The proposed method significantly improves the performance when compared to the baseline solutions and is able to identify parameters that can be used to distinguish between the gait abnormalities and help therapists provide targeted treatment in clinics. In the third part,
we present a new approach for identifying unique gait patterns, that can be attributed to unique individual behavior, and provide gait-based biometric authentication. Wearable sensors such as smart shoes and socks are used as gait sensing devices and are capable of recording acceleration and ground contact forces. The proposed approach relies on multimodal learning, with a neural network of bimodal-deep auto-encoders. The proposed methodology outperforms existing solutions, and provides robust and user friendly mobile authentication experience.
An Algorithmic Framework For Gait Analysis and Gait-Based Biometric Authentication

Ioannis Papavasileiou

B.Eng., University of Patras, Greece, 2011

A Dissertation
Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy at the University of Connecticut

2018
APPROVAL PAGE

Doctor of Philosophy Dissertation

An Algorithmic Framework For Gait Analysis and Gait-Based Biometric Authentication

Presented by

Ioannis Papavasileiou, B.Eng.

Major Advisor

Dr. Song Han

Associate Advisor

Dr. Wenlong Zhang

Associate Advisor

Dr. Jinbo Bi

University of Connecticut

2018
To my father
ACKNOWLEDGMENTS

This dissertation would not have been successful without the help and support from several people. First and foremost I would like to thank my advisor Prof. Song Han, who has helped me and guided me throughout all the lengths and widths of my PhD research. It has been an honor to work with him and his continuous inspiration, challenging questions and unlimited guidance enabled me to achieve my research goals and become an independent researcher.

I would also like to thank Prof. Wenlong Zhang for working with me throughout all my Ph.D. years and the discussions we had. Thanks to his multidisciplinary knowledge in sensing technologies and connected health, I have benefited greatly from his input in multiple areas. In addition, I would like to thank Prof. Jinbo Bi for her continuous feedback and valuable input from her expertise in Machine Learning. Without her contribution, I would have never been able to complete the work in this dissertation.

I would like to thank Mr. Zhi Qiao, Mr. Wenhao Deng, Dr. Xin Wang, and Ms. Savana Smith who I had the honor to collaborate with during the years of my Ph.D. research, for their inspiration on research ideas, contribution in developing different parts of the infrastructure and algorithms used in this dissertation, and conducting of human participant studies.

I would like to thank the members of my research group Tao Gong, Shaobo Zheng, Peng Wu, Zelin Yu, Areej Khudhran Althubaity and Huayi Ji for their fruitful discussions and great support. I would like to thank all the friends that I had the honor to meet during all this time, Misagh Kordi, Theophanis Hadjistasi and Aaron Palmer. Their support, friendship and advise made my graduate school years a great experience.

Finally I would like to thank my parents Stavros and Eleni, my brother Thanassi and all my family for their endless support, love and understanding through the years.
Contents

1 Introduction .......................................................... 1
   1.1 Target applications .............................................. 3
       1.1.1 Objective gait rehabilitation ............................. 4
       1.1.2 Gait-Based Biometric Authentication ...................... 6
   1.2 Contributions ................................................... 7
   1.3 Thesis Outline .................................................. 9

2 Related Works ...................................................... 10
   2.1 Sensing platforms .............................................. 10
   2.2 Gait Phase Detection ........................................... 11
       2.2.1 Gait Cycle and Gait Phases ............................... 11
       2.2.2 Gait Phase Detection Algorithms ......................... 12
       2.2.3 Cloud-based Healthcare Applications ...................... 13
   2.3 Disorder Diagnosis ............................................. 14
       2.3.1 Gait quantification ....................................... 14
       2.3.2 Gait pattern classification .............................. 15
   2.4 Gait-Based Biometric Authentication .......................... 16

3 Real-time Data-driven Gait Phase Detection using Ground Contact Force Measure-
## 3.1 Gait Phase Detection Using Infinite Gaussian Mixture Model

### 3.1.1 Smart Shoes for GCF Measurement

### 3.1.2 Finite and Infinite Gaussian Mixture Models

## 3.2 Parallel Particle Filter Design

### 3.2.1 An Overview of Particle Filter Method

### 3.2.2 Design Principles of the Parallel Particle Filter

### 3.2.3 Particle Sharing Mechanisms

### 3.2.4 Algorithm Details

## 3.3 System Implementation

## 3.4 Performance Evaluation

### 3.4.1 Experimental Settings

### 3.4.2 Estimation accuracy of the gait phase detection algorithms

### 3.4.3 Computational efficiency of the proposed methods

### 3.4.4 Determining number of particles

## 3.5 Conclusion

## 4 Classification of Neurological Gait Disorders Using Multi-task Feature Learning

## 4.1 Gait Features Extraction

### 4.1.1 Gait Cycles

### 4.1.2 Gait Phase Detection

### 4.1.3 Gait Phase Features

### 4.1.4 Mobility Features

### 4.1.5 Balance and Strength Features

## 4.2 Multi-Task Feature Learning for Gait Disorder Diagnosis

## 4.3 Performance Evaluation

### 4.3.1 Human Subject Test Design

### 4.3.2 Classification of Gait Disorders
4.3.3 Identification of Important Gait Features ........................................ 69
4.4 Conclusion .......................................................................................... 73

5 Gait based authentication ...................................................................... 75

5.1 Methodology overview ........................................................................ 78
5.2 Data Acquisition Platforms ................................................................. 80
5.3 Data Filtering and Gait Cycle Detection ................................................ 83
  5.3.1 Filtering ......................................................................................... 83
  5.3.2 Gait Cycle Detection ...................................................................... 84
5.4 Feature Extraction and Classification of Gait Patterns ........................ 85
  5.4.1 Feature Extraction with Auto-encoders ......................................... 86
  5.4.2 Early and Late Sensor Fusion Techniques ..................................... 87
  5.4.3 Classification of Gait Features ....................................................... 88
5.5 Performance Evaluation ...................................................................... 90
  5.5.1 Case Studies .................................................................................. 90
  5.5.2 Experimental Setup ....................................................................... 92
  5.5.3 Expt. 1: Cross-validation and Leave-one-out evaluation ............... 94
  5.5.4 Expt. 2: Evaluation of Parameters that Affect Gait ....................... 96
  5.5.5 Expt. 3: Evaluation on the Impact of Training Time ..................... 100
  5.5.6 Expt. 4: Active Attacks through Gait Mimicking ......................... 101
  5.5.7 Summary and Discussion of the Experiment Results .................... 102
5.6 Conclusion ........................................................................................... 104

6 Conclusions and Future Work ............................................................... 105

Bibliography ......................................................................................... 108
# List of Figures

1.1 An overview of methodology design for gait quantification, analysis and pattern recognition ........................................... 4

2.1 An overview of a gait cycle and the gait phases ......................................................... 12

3.1 An overview of the smart shoe design. A signal processing unit includes barometric sensors, microcontroller, and Bluetooth chip ......................................................... 22

3.2 GCF data from PD and post-stroke patients and a healthy subject ......................................................... 24

3.3 An overview of the parallel particle filter ......................................................... 31

3.4 An overview of the real-time data analytics platform for gait physical therapy monitoring and evaluation ......................................................... 38

3.5 The topology of the parallel particle filter on Apache Storm ......................................................... 39

3.6 (a) Raw GCF measurements from a healthy subject, (b) gait phases detected from the particle filter algorithm, (c) gait phases detected from fuzzy logic ......................................................... 42

3.7 (a) Raw GCF measurements from a PD patient, (b) gait phases detected from the particle filter algorithm, (c) gait phases detected from fuzzy logic ......................................................... 43

3.8 (a) Raw GCF measurements from a post-stroke patient, (b) gait phases detected from the particle filter algorithm, (c) gait phases detected from fuzzy logic ......................................................... 44

3.9 The distribution of the execution time of the gait phase detection algorithm running with varied workers for different patient classes ......................................................... 48

3.10 Fault rate and execution time vs. particle numbers ......................................................... 50
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.11</td>
<td>Gait phases detected when 20 particles were used</td>
<td>51</td>
</tr>
<tr>
<td>4.1</td>
<td>An overview of a gait cycle and the gait features from four categories: gait phases, mobility, balance and strength.</td>
<td>56</td>
</tr>
<tr>
<td>4.2</td>
<td>Feature selection vector $c$ from all MMTFL methods</td>
<td>71</td>
</tr>
<tr>
<td>4.3</td>
<td>Absolute value of task parameter vector $\alpha_t$ in the Stroke vs. healthy gait classification task.</td>
<td>72</td>
</tr>
<tr>
<td>5.1</td>
<td>An overview of the proposed integrative framework for multimodal gait-based continuous authentication: (a) sensing platforms, (b) filtering/cycle extraction, (c) feature extraction, (d) classification, and (e) authentication.</td>
<td>78</td>
</tr>
<tr>
<td>5.2</td>
<td>The two sensing platforms for data acquisition</td>
<td>81</td>
</tr>
<tr>
<td>5.3</td>
<td>Raw ACC and GCF data from both the smart shoes and Sensoria socks.</td>
<td>82</td>
</tr>
<tr>
<td>5.4</td>
<td>The conceptual structure of the autoencoder</td>
<td>86</td>
</tr>
<tr>
<td>5.5</td>
<td>Overview of the models used for sensor fusion and classification</td>
<td>89</td>
</tr>
<tr>
<td>5.6</td>
<td>Data collection environment for the smart shoes case study</td>
<td>91</td>
</tr>
<tr>
<td>5.7</td>
<td>10-fold and leave-one-out CV for both socks and shoes studies</td>
<td>96</td>
</tr>
<tr>
<td>5.8</td>
<td>Per-subject leave-one-out performance</td>
<td>97</td>
</tr>
<tr>
<td>5.9</td>
<td>A summary of the results from all testing scenarios in Table 5.1</td>
<td>99</td>
</tr>
<tr>
<td>5.10</td>
<td>The impact of training time on the authentication performance</td>
<td>101</td>
</tr>
</tbody>
</table>
## List of Tables

4.1 Proposed twelve gait features in four categories .................................................. 55

4.2 AUC performance of different methodologies ................................................. 67

4.3 Per task average AUC scores when a new subject is tested in a model trained by the rest subjects .................................................. 67

4.4 Confusion Matrices of $\text{MMTFL}\{1,2\}$ for the 3 tasks, true labels in rows, predicted in columns .................................................. 68

4.5 Confusion Matrices of $\text{MMTFL}\{2,1\}$ for the 3 tasks, true labels in rows, predicted in columns .................................................. 68

4.6 Confusion Matrices of $\text{MMTFL}\{2,1\}$ for the 3 tasks per patient .................. 69

5.1 A summary of the two case studies with different sensing platforms and testing scenarios .................................................. 91

5.2 EER and FAR performance per modality and fusion model, with SVM classification .................................................. 94

5.3 A summary of the experimental results on the performance of the raw-stacked model. M and A refer to Morning and Afternoon recording sessions, respectively. $\mu$ and $\sigma$ are the mean and standard deviation of the reported metric, respectively. .................................................. 100

5.4 FAR scores with gait mimicking under different models. Subject 6 is the attacker and subject 9 is the victim. .................................................. 102
Chapter 1

Introduction

In the past decade, we have seen tremendous advances in sensing and wearable technologies, which has enabled the recording and monitoring of motion, physiological, neurological and biochemical measurements [85]. With further developments in biofeedback [22, 96, 97] and assistive technologies [20], we have seen great improvements in multiple healthcare applications. Moreover, smartphones have been embedded with advanced and diverse suites of sensor components and due to their increasing wireless connectivity [24] and computation capabilities, now they act as sync devices for body-area networks [32] [45] [46] and wearable systems, which has enabled monitoring of human activity [121]. These advancements, in combination with developments in cyber-physical systems [98], communication technologies [24], and cloud computing infrastructures, have resulted in an explosion of recorded sensing and contextual data, and thus increase in available information that was never been possible before.

With this progress in sensing, increased availability of diverse and rich information and with advances in computational methods, artificial intelligence and machine learning, we observe a development of a plethora of applications in multiple domains of our lives, including healthcare, mobile and ubiquitous computing. This makes it possible to develop smart systems to monitor activities of humans continuously at home or in clinics, and thus new opportunities have raised in multiple applications, such as gait physical therapy, human activity recognition, behavior moni-
With the increasing use of sensing and assistive technologies in gait physical therapy, instrumented treadmill and wearable sensors have been developed to measure force, torque, and kinematics during walking to facilitate disease diagnosis and training plan development, during gait physical therapy [84]. Based on the collected sensor data, various biofeedback mechanisms have been developed to make the data intuitive and helpful for patients and medical professionals [37]. However, the current clinical gait training is primarily carried out by the physical therapists, who observe patients’ walking patterns and use clinical measures to design the training plan, and provide active assistance and stimulation to help patients regain walking capability. These standard clinical approaches cannot fully satisfy the needs from patients, as they are labor-intensive, subjective, and expensive. Moreover, patients have to visit clinics regularly and only get treatment during training sessions. This is inconvenient and time-consuming, and the patients are unable to exercise at home and receive feedback from the therapists, which significantly slows down the rehabilitation progress [11]. Therefore, there is an increasing need for automating processes in gait analysis and rehabilitation.

With the rise in popularity of wearable technologies and smartphones, we have also seen great improvements in human activity recognition, and behavior monitoring applications [102]. This has led to new discoveries for gait patterns recognition, with applications in ubiquitous interactive and multi-modal computing. New methods have been developed that can identify different types of information from gait patterns, including information related to the behavior of the individual, such as drinking behaviors [51], or their mood, affective and emotional state [14] [63]. In addition, gait patterns sometimes may be understood by others as something beautiful and may convey attractiveness and health, which has led to development of methods for identification of beautiful gait [75].

All these recent developments in gait pattern recognition and behavior detection have led to the rise of biometrics, which can improve significantly security and usability, especially in mobile phone devices [112]. A special case of that is gait-based behavioral biometrics, which
can be used to identify an individual’s identity or authenticate a device owner only based on their individual gait patterns [66, 124]. Despite these great developments, many solutions for gait-based biometric authentication have been based on data recorded from inertial measurement units (IMU), and a common issue of IMU measurements is the lack of accuracy, robustness and reliability [100].

Motivated by the increasing demand for automating processes in gait analysis and rehabilitation, the different amount of information that can be carried in gait patterns and the need for improved gait patterns recognition for biometric authentication, in this thesis we present an algorithmic framework that is designed to identify gait patterns which can help facilitate gait analysis, provide objective gait patterns recognition for gait disorder diagnosis and identify gait patterns that are attributed to the individual characteristics of every human, to further improve effectiveness and robustness of gait-based biometric authentication. In order to better understand the motivating applications behind the development of this framework, we first discuss possible target applications that may benefit from such an advanced framework for gait analysis and biometric authentication. We then summarize the contributions made with all the components of this thesis, and finally we conclude the introduction with the outline of this thesis.

1.1 Target applications

The proposed algorithmic framework in this thesis aims to provide solutions to those applications that require different levels of resolution from the common problem of gait pattern classification or categorization into groups. First, we discuss the target applications in gait rehabilitation, which includes problems such as gait phase detection and gait disorder diagnosis. Then we discuss another target application of the proposed framework, i.e. gait-based biometric authentication.
1.1.1 **Objective gait rehabilitation**

Any dysfunction of the central nervous system, spinal cord, peripheral nerves or muscles can result in an abnormal gait [28]. Additionally, aging is another factor that can affect gait patterns. At the age of 60, 85% of people have a normal gait, while at the age of 85 or older this proportion drops to 18% [99]. As a result, an increasing number of people suffer from walking difficulties, and the demand for gait rehabilitative therapy has been increasing rapidly in the past few years.

Gait quantification is important for objective gait assessment, analysis and diagnosis. It relates to the methods used for objective estimation of gait cycles and gait phases, and measurement of gait parameters which can be used to assess the severity of a subject’s gait abnormality. Gait analysis is the systematic examination of the way in which a person walks [116]. It may be conducted either for clinical purposes or for research. In the clinical area, it may be used for diagnosis, assessment, or for monitoring the results of treatment. An overview of the methodology design for gait quantification, analysis and pattern recognition can be seen in Fig. 1.1.
Despite the massive development of sensing systems, gait analysis still requires significant input from therapists and cannot be conducted automatically. There are multiple challenges, with the first one being lack of qualified therapists given the fast increasing number of patients. Moreover, therapists need to spend a lot of effort to fully understand the sensor signals and perform rapid gait evaluation, which makes such sensing systems less applicable in an actual clinical environment. In addition, the input from therapists will be subjective and introduce inconsistencies into the evaluation results.

The first part of the proposed framework in this thesis will focus on the problem of real-time gait phase detection, which is essential for gait rehabilitation because it enables the patients to identify their walking abnormalities and make corrections immediately, as it can be used in providing real-time feedback to the training of patients. This component is the foundation of the proposed framework, as more advanced information can be extracted once gait phases have been identified objectively for both healthy subjects and patient populations.

To better quantify the severity of abnormal gait, important sensing features need to be identified from the sensory data to characterize gait disorders. Towards this goal, extensive research efforts have been reported to use machine learning algorithms for gait classification and clustering, to identify such parameters and automate gait disorder diagnosis. For example, post-stroke patients usually experience a very diverse set of gait abnormalities, most common of which is the hemiplegic gait [28]. For this reason, researchers have applied cluster analysis to identify subgroups of patients with similar sensing features who experience similar gait abnormalities [23, 52, 69]. Likewise, other research efforts focus on classifying abnormal gaits between healthy subjects and Parkinson’s disease (PD) patients [58, 106, 118]. Classification methods with feature selection can help the target design of treatment and evaluation of therapy through the identified important gait sensing features [19]. Furthermore, such tools can improve the valuable clinical management of the patients, ease communication between clinicians [19] and optimize subject selection for human participant studies [34]. Consequently, they reduce the cost of physical therapy and improve the quality of life for patients. Especially, patients living in remote areas
can benefit from an enhanced tele-medicine system with these quantitative and diagnostic tools, without necessitating complex apparatus [19]. However, to the best of our knowledge, there is no such quantitative gait diagnostic system for neurological diseases. In the second part of this thesis we propose a novel methodology for automatic diagnosis of gait abnormalities and disorders from extracted gait parameters. The proposed algorithm is able to provide very accurate prediction of underlying abnormalities, and identify important parameters that are used for this decision. This can help therapists identify gait parameters that are affected by each disorder and improve the targeted treatment.

1.1.2 Gait-Based Biometric Authentication

Gait patterns convey a considerable amount of information about the individual and it is claimed that gait is fairly unique to them due to one’s specific muscular-skeletal structure [125]. As such, gait-based authentication is among the most popular behavioral biometric authentication methods. Gait-based continuous authentication utilizes such characteristics from an individual’s gait in real time. It requires little cooperation from the user, and is usually an inexpensive option. Gait is also difficult for an adversary to mimic [36, 65, 66], which makes hacking gait based authentication hard. For example, fake fingers have been used to get access in fingerprint based systems, recorded voice has been used in voice recognition systems, and pictures and masks have been used for face recognition based authentication systems [36, 85].

Many sensing platforms have been used for gait-based authentication, such as cameras for computer vision-based gait recognition [21, 42], or smart mats [82] and plates [30] for floor sensing. However, wearable sensors and mobile phones are more popular platforms due to their mobility, small size and low cost. Most of the wearables or mobile phones being used in gait-based authentication research are equipped with an inertial measurement unit (IMU). However, a common issue of IMU measurements is the lack of accuracy, robustness and reliability [100]. This can be partially attributed to the high sensitivity to the location and orientation of the sensor.
and increased level of noise present in the recorded data. To reduce such effects, multimodal learning and sensor fusion techniques have been introduced in some recent research developments [124]. In order for such methods to be more accurate, a number of IMU devices have been deployed at multiple places of the human body, to build a stronger model based on weaker sources.

The goal of the proposed algorithmic framework in this thesis is to provide solutions to all these target applications. With the development of this framework, multiple contributions have been made, which are discussed in the following section.

1.2 Contributions

• In order to achieve intelligent gait analysis based on sensor signals, different sensor fusion and machine learning algorithms have been employed, such as fuzzy logic, hidden Markov models (HMM), support vector machine (SVM), and so on. Most of these algorithms, however, are computation-intensive. They require significant computational resources and cannot run in real-time during rehabilitation training. This poses a new challenge for the successful implementation of these algorithms for real-time gait analysis. On the other hand, the rapid development of big-data driven analytic techniques have been observed in many healthcare applications. It provides a new solution to fully enable the processing of massive sensory data for complex decision making and analysis. Motivated by this trend, the first component of the proposed framework in this thesis focuses on the design of a data-driven approach for real-time automatic gait phase detection algorithm, and implement it on a cloud-based gait monitoring and analysis platform. Our solution combines an infinite Gaussian mixture model (IGMM) to classify different gait phases based on the GCF measurements, and a parallel particle filter to estimate and update model parameters. The parallel particle filter is abstracted as a computation topology and deployed on a parallel real-time computing framework in a Microsoft Azure cluster [3]. Both time-based
and weight-based particle sharing mechanisms are also proposed to judiciously distribute particles among different working nodes and thus strike a good balance between computational overhead and estimation accuracy. The effectiveness of the proposed algorithm is validated using the data traces collected in a clinical study from five PD patients, three post-stroke patients, and three healthy subjects. Significant improvements on both computational efficiency and classification accuracy have been observed.

• In order to enable objective gait analysis and to automate the diagnostic gait assessment, we also propose to enhance our gait analysis framework with an integrative set of components to automatically classify gait disorders from two common neurological diseases, stroke and PD, and distinguish abnormal gait caused by these two diseases from the healthy gait. Classifying gait into groups caused by these two major neurological diseases can lead the way to provide diagnostic tools for specific gait disorders caused by these two neurological diseases, which is much needed for assisting objective gait assessment in the clinic and rehabilitation therapy centers. Our integrative framework includes a pair of smart shoes as the sensory device to capture the GCF data and a pipeline of data analytic algorithms for feature extraction, classification and feature selection. Gait features, including mobility, balance, strength and rhythm, are extracted from the sensory data.

• Finally, in order to further extend the proposed framework to recognize gait deviations from not only patients suffering from neurological disorders, but also healthy adults, we propose a gait-based continuous authentication method which uses multimodal learning. Specifically our approach aims to support a more user friendly and robust authentication method with the use of two sensing modalities, i.e., accelerometer (ACC) data and ground contact forces (GCFs). We employ a multimodal learning approach based on deep autoencoders to explore the correlations between these two different modalities of the data and thus build more robust learning models leading to more accurate authentication results. Two types of sensor fusion techniques are explored, i.e. early and late sensor fusion.
Early fusion is based on the hypothesis that it is possible to develop models that use simple
time-domain features for authentication, while the hypothesis for late sensor fusion is that
more complex and abstract features are required for gait-based authentication, and thus
extraction of higher-order features based on simple time-domain features is required.
The effectiveness of our approach is evaluated through extensive experiments on datasets
collected from two case studies, one with commercial off-the-shelf (COTS) smart socks
and the other with a research prototype of smart shoes, both of which can record GCF and
ACC data. Based on the collected datasets, we evaluated the robustness of the proposed
methodology under different scenarios, such as a per-modality evaluation of authentication
performance, generalization capability to a new impostor that has not provided training
samples, effects of different walking conditions (e.g., slow-fast walking), and changes of
gait patterns due to the effect of daily energy level change from morning to afternoon. Our
extensive experimental results show that the proposed approach can significantly increase
the robustness of gait-based authentication with the introduction of multimodal learning
based on autoencoders.

1.3 Thesis Outline

Chapter 2 presents a literature survey of related works to this thesis. Chapter 3 presents an objec-
tive real-time gait phase detection algorithm that does not need parametrization from the physical
therapists [80]. Based on that, chapter 4 discusses our gait disorder diagnosis algorithmic fram-
work, that can improve the treatment provided to the patients, as targeted and personalized plans
can be built based on the gait parameters that affect the patient’s gait [81]. Chapter 5 presents the
algorithmic framework for gait based authentication, which relates information from multiple
modalities to improve its efficiency. This thesis is concluded in chapter 6 with a discussion of
what has been accomplished in this dissertation and a discussion on future directions that we are
looking to pursue to further extend this work.
Chapter 2

Related Works

Walking is a basic requirement for many of the human basic daily activities and the ability to walk safely, effectively and efficiently is essential for an independent and productive life. Gait is the manner or style in which a locomotor activity, such as walking or running, is undertaken [59]. All voluntary movement, including walking, results from a complicated process involving the brain, spinal cord, peripheral nerves, muscles, bones and joints [59], which makes gait one of the most universal and complex of all human activities. In order to facilitate gait analysis and develop biometric authentication methods, multiple components are required. In this section we discuss background and related works on all the components of the proposed framework for gait analysis and biometric authentication, including sensing platforms, gait phases detection, gait disorder diagnosis, and gait-based biometric authentication.

2.1 Sensing platforms

In order to provide improved methods for gait analysis and biometrics, better sensing platforms are needed. Significant research efforts have contributed in the development of multiple sensing platforms and wearable devices, which have significantly helped in providing more objective gait assessment. A variety of sensory devices have been employed for recording of gait pat-
terns. For instance, encoders, inertial sensors, and camera-based motion capture systems have been employed for kinematic analysis of human motion [57, 93]; force sensors [12, 60] and electromyography (EMG) sensors [104] have been widely used to study the ground contact forces (GCFs) and muscle activities during walking; electroencephalography (EEG) sensors have been employed to analyze brain signals [41, 92] and better understand neurological mechanisms of walking.

2.2 Gait Phase Detection

In this section, we first give an overview of a gait cycle and its associated gait phases, followed by a review of existing methods for gait phase detection. We then summarize recent development of cloud-based healthcare applications.

2.2.1 Gait Cycle and Gait Phases

Gait cycle is the time interval between the same repetitive events of walking. The defined cycle can start at any moment, but generally begins when one foot contacts the ground. If it starts with the right foot contacting the ground, the cycle ends when the right foot makes contact again. Fig. 2.1 gives an overview of two gait cycles at the lower two horizontal solid lines. The gait cycle can be broadly divided into two phases: stance phase and swing phase [28]. These two phases can then be further divided into sub-phases within the gait cycle, as shown at the top part of Fig. 2.1. In general, the stance phase takes around 60% of the gait cycle [28] and can be divided into double support and single support. In double support, both feet are in contact with the ground, while in single support only one foot is in contact with the ground. Double or single support ratio refers to the portion of time within a gait cycle someone spends in double or single support respectively. The swing phase is described when the limb is not weight bearing and represents around 40% of a single gait cycle [28]. These percentages can change with the
walking speed, as with a higher speed the double support ratio in the gait cycle tends to be reduced. In Fig. 2.1 the lower depicted cycle starts with right foot initial contact, which leads to the stance phase, while the other starts with left pre-swing phase which leads to swing phase. Indicative percentages are shown to indicate the different phases within the cycle.

Gait phases are shown at the top of Fig. 2.1 and they refer to various states within one walking cycle. There are typically eight gait phases for a healthy subject (as shown at the top of Fig. 2.1): initial contact, loading response, mid-stance, terminal stance (or initial contact), pre-swing, initial swing (not shown in Fig. 2.1), mid-swing, and terminal swing [54, 122]. However, in a pathological gait, some gait phases might be missing and the time allocation of gait phases might also be different from a normal gait. This provides a powerful tool for abnormal gait detection and evaluation of rehabilitation training performance. Given the smart shoes used to collect data for this work (see Section 3.1.1) are not equipped with rotation angle sensors, our design is only able to detect one distinct swing gait phase, instead of three.

2.2.2 Gait Phase Detection Algorithms

Based on the specific objective of gait phase detection and information contained in the sensor signals, various sensors and algorithms have been developed to classify all or some gait phases. The sensors used in gait phase detection include force sensors, inertial sensors, EMG sensors,
Electroneurography (ENG) sensors, and ultrasonic sensors [105]. For example, one popular type of sensing devices is smart shoes or insoles embedded with sensors to measure GCFs [10, 54, 88]. Fuzzy logic rules [54], hidden Markov models [10], and neural network [48] have been proposed to detect gait phases based on the GCF signals. Inertial sensors provide another solution for detecting gait phases based on the measured joint kinematics [56, 122]. Various algorithms have been developed including threshold-based rules [56], hidden Markov models [62], and SVM [107]. ENG sensors have been employed in combination of a Gaussian mixture model (GMM) to distinguish stance and swing phases [25]. However, all the algorithms above require subjective input from both therapists and engineers. For example, the number of expected gait phases need to be pre-defined, which is very difficult especially for patients. The threshold values of fuzzy logic rules and threshold-based algorithms need to be decided subjectively and they need to be adjusted for different subjects.

2.2.3 Cloud-based Healthcare Applications

Cloud-based healthcare applications are becoming pervasive. Large volumes of real-time data from both patients and their living environments are captured and analyzed for close health monitoring and adverse event prediction [90]. Among these many applications, wireless body area networks are connected to the cloud and used for monitoring physiological sensor data in both home and hospital [31]; surveillance systems are combined with cloud-based computing platforms for disease outbreak and medical condition predictions [16, 111]. From the analytics aspect, significant research efforts have been made on algorithm development for clustering, classification, frequency counting, time series analysis and data streams mining and processing [35, 68, 127]. For all these solutions, there is however a great need of a scalable real-time cloud platform that optimizes the resource allocation to provide real-time data analytics [120].
2.3 Disorder Diagnosis

Extensive research efforts have been made towards quantitative gait analysis. In this section, we first discuss the literature studies on improving gait quantification methods for objective gait parameter extraction. We then present a summary on machine learning methods for improving gait analysis, which includes gait pattern classification and cluster analysis for finding subgroups of patients who suffer from the same neurological disease and experience similar gait abnormalities.

2.3.1 Gait quantification

Gait quantification is important for objective gait assessment and analysis. It relates to the methods used for objectively measuring gait parameters, which can be used to estimate the severity of human gait abnormality. In this subsection we discuss gait quantification with respect to hemiplegic and Parkinsonian gait, which are the two most popular gait disorders caused by stroke and PD respectively [28].

Among many gait parameters, symmetry is an important gait characteristic and is defined as a perfect agreement between the actions of the two lower limbs [91]. To calculate symmetry, mobility parameters (e.g., single support ratio) or spatiotemporal parameters (e.g., step length) can be used [86].

Balance or walking stability is another important parameter that needs to be quantified, and used to predict falls. In [44] multiple balance and stability measures are proposed, including RMS acceleration, jerk (time series of first derivative of acceleration), sway (a measure on how much a person leans his/her body), step and stride regularity and variability. Mobility and gait phases are also important gait parameters used to quantify gait. Mobility parameters include general movement characteristics like cadence, step length, single and double support ratio and periodicity [64, 86]. Gait phases refer to the various states within one walking cycle, and there are typically eight gait phases for a healthy subject [80].

Gait quantification can be used to extract gait features for gait pattern classification. In this
thesis we calculate standard gait parameters based on GCF data for mobility, balance and strength quantification. In addition, new gait phase parameters are introduced based on our previous work [80, 122], in which a wireless human motion monitoring system was designed, and a real-time data-driven gait phase detection algorithm was developed to capture the gait phases based on the recorded GCF data. The proposed system can objectively quantify the underlying gait phases without any input from a medical professional. These two works lead to some of the gait parameters used in the second part of this thesis.

2.3.2 Gait pattern classification

Extensive research efforts have been reported to perform cluster analysis of post-stroke gait patterns and enable targeted treatment. In [69] non-hierarchical cluster analysis was used to categorize four subgroups based on the temporal-spatial and kinematic parameters of walking. Similarly, hierarchical cluster analysis of post-stroke gait patterns was conducted in [52], identifying three groups of patients with homogeneous levels of dysfunction. In [34], k-means clustering was used to group gait patterns in order to optimize participant selection for a biofeedback pedaling treatment.

Classification of post-stroke and PD gait patterns is another example of using machine learning methods in gait analysis. Classification of post-stroke gait patterns against healthy gait was performed in [23] and [64], using kinematic and kinetic data. Artificial neural networks (ANN) were used in [49] to classify post-stroke patient’s gait into three categories based on the types of foot positions on the ground at first contact: forefoot, flatfoot, and heel. The work in [19] classified hemiparetic gait in three groups with two subgroups each, that were defined from clinical knowledge. This classification method had the advantage of great usability in clinical routines without necessitating complex apparatus. Classification of PD gait patterns against healthy gait is also studied [58, 106, 118]. Gait features from wavelet analysis and kinematic parameters are extracted, which are passed to support vector machines (SVM) and artificial neural networks.
In the second part of this thesis (Chapter 4), we perform classification of gait patterns in three classes, healthy, Parkinson’s and post-stroke. To the best of our knowledge, there is no research work on classification of gait patterns between these three classes. We employ a comprehensive set of gait parameters - including mobility, balance, strength and gait phases - and send them as input features to a classifier. An advanced classification method, MTFL, is used to distinguish between the three gait classes.

### 2.4 Gait-Based Biometric Authentication

Gait-based behavioral biometrics can be used in both identification and authentication scenarios [125]. For identification, a sample gait is compared to a database of enrolled gait samples with known identities to determine whom the unknown sample belongs to; for authentication, a gait sample is compared to the enrolled sample gait data for a known person to validate his or her identity. Gait biometrics have attracted tremendous research attentions in recent years due to two main reasons: the rapid development of sensing technologies on mobile devices, and the increasing popularity and usability of biometric authentication compared to traditional authentication methods. Apart from that, gait-based authentication may be harder to spoof when compared to other widely used physiological biometric methods. For example, fake fingers have been used to get access in fingerprint based systems; recorded voice has been used in voice recognition systems; and pictures and masks have been used for face recognition based authentication systems [8, 36, 85]. In addition, gait-based biometric authentication may be preferred from individuals when compared to other biometric methods which may be considered more privacy intrusive, e.g. face recognition that requires the use of cameras [112].

In general, there are three main approaches for gait-based authentication, i.e., computer vision techniques, sensing on the floor with smart mats and plates, and wearable devices. Computer vision techniques are based on video recordings of the subject to be authenticated or identified.
They have drawn great attention recently due to their promising application to security, monitoring and surveillance systems in public places, such as airports. Floor sensing has been performed with the use of smart mats [82], force plates [30] and floor vibration measurement [79]. Its application mostly focuses on the identification of people entering a restricted area.

Using wearables to perform gait-based biometric authentication is another popular approach due to their mobility, small size and low cost. That includes the use of IMUs and devices such as smart phones and smart watches. These wearables can record signals at multiple locations on the human body, including wrist [119], waist [72] [100], breast pocket, trouser pocket [29, 47, 125, 126], and hip [36, 74]. A summary of recent research work on using accelerometer for gait recognition can be found in [124] and [101]. A common issue of IMU measurements is the lack of accuracy, robustness and reliability [100]. This can be partially attributed to the high sensitivity to the location and orientation of the sensor and increased level of noise present in the recorded data. However, many of these recent research efforts have achieved remarkable performance. For example, a 0.8% equal error rate (EER) was reported in [103] by applying a curve aligning approach on the dataset collected from 22 subjects. Research also shows that EER keeps increasing when the dataset size grows, e.g. [100] reports an EER of 6% to 12% evaluated on a large open source dataset of 744 subjects. This increase may be related to multiple other parameters on top of the population size, such as types of sensing technologies, sensor placement and noise levels in the data.

Besides wearables, smart phones can also be easily used as sensing devices for gait recognition, as they require no additional hardware support [61]. However, the effectiveness of using smart phones for gait authentication heavily depends on the location and orientation of the phone to be deployed on the human body. The performance will further degrade when the subject performs everyday actions with their phones (e.g., making calls or browsing the web [115]). Among the many research attempts in this direction, [95] studies gait authentication based on multiple placements of phones on the human body; several work rely on using the phones in a fixed position in their pocket [73]; [125] addresses the issue of variable phone orientation by computing
invariant gait representations and uses gait dynamic images to extract features, which achieves an EER of 3.88-7.22% with 55 human subjects; [47] presents a pace-independent gait identification system with 36 subjects, achieving verification rates (VR) of 61.1-99.4% with a False Acceptance Rate (FAR) of 0.1%.

To further improve the authentication/identification performance and alleviate the performance degradation from noisy measurements, multimodal methods and sensor fusion techniques have been used to combine different types of data from multiple sources [124]. These approaches are attractive as they can effectively relate the increased information available from multiple sources and modalities and thus result in better models that outperform the traditional methods. For example, [26] introduces a continuous authentication method for mobile devices based on fusion of face images and IMU data. [124] presents a sparse representation method with the use of four accelerometer data sources, achieving an EER of 2.2% for verification. Such approaches however may come with reduced usability as the potential user may need to wear multiple wearable devices, making the setup not practical for everyday use. In addition, most of the studies fail to report leave-one-out cross-validation, which gives a better estimate of the generalizability of the approach when a new impostor subject is tested whose gait data are not used to train the corresponding model. For the studies that do report leave-one-out cross-validation, the best FAR performance achieved is 3% with 11 subjects [109] and 6% with 32 subjects [110].

In this work, we use two modalities for gait-based authentication, which combine accelerometer (ACC) and ground contact force (GCF) data. GCF measurements can be recorded with the use of smart socks or smart shoes, which has recently seen great advancement in multiple domains, especially for gait rehabilitation [54, 55, 122]. We hypothesize that by building models that combine data from multiple sources, we can achieve robust gait-based authentication. To the best of our knowledge, this is the first attempt in gait-based authentication using a combination of ACC and GCF data. Improved usability can be achieved with this approach, as users do not need to use extra wearables, except from a pair of smart socks or smart shoes that can record both modalities. With the rapid development of new wearable technologies, we envision that
most shoes and socks in the near future will be equipped with smart sensors to capture ACC and GCF data in a continuous manner.
Chapter 3

Real-time Data-driven Gait Phase Detection using Ground Contact Force Measurements

The world is experiencing an unprecedented demographic shift. According to the report from U.S. Department of Commerce [78], more than 20 percent of the U.S. residents are projected to be aged 65 or over by 2030, compared with 13 percent in 2010 and 9.8 percent in 1970. Aging results in changes in memory, balance, and mobility for healthy subjects, and it is also associated with increased rates of degenerative conditions of the musculoskeletal system, cardiovascular system, and most importantly, the nervous system (e.g., Alzheimer’s disease (AD) [1], stroke [7], and Parkinson’s disease (PD) [2]). One major consequence of neurological diseases is gait disorders, and consequently the demand for gait rehabilitation has increased rapidly over the years.

Current gait rehabilitation is provided by physical therapists who make gait evaluation based on their clinical experience and employ manual techniques to train the gait patterns of patients [17]. As the first step to improve the accuracy of gait evaluation, various sensory devices have been developed to collect biosensing data from patients for gait analysis and impairment
To name a few, pressure and force sensors have been adopted to measure the ground contact forces (GCFs) for gait inference [54, 60]; electromyogram (EMG) sensors have been used to analyze the muscle activities during walking [123]; inertial sensors have been widely used to estimate the walking kinematics of lower extremity [83]; vision-based sensors have also been employed for reconstruction of human motion [93]. With these biosensing data, physical therapists can gain a better knowledge of the patients’ gait behaviors and make appropriate training plans.

Despite the massive development of sensing systems, gait analysis still requires significant input from therapists and cannot be conducted automatically. There are multiple challenges, with the first one being lack of qualified therapists given the fast increasing number of patients. Moreover, therapists need to spend a lot of effort to fully understand the sensor signals and perform rapid gait evaluation, which makes such sensing systems less applicable in an actual clinical environment. Last but not least, the input from therapists will be subjective and introduce inconsistencies into the evaluation results. In particular, this chapter will focus on the problem of real-time gait phase detection, which is essential for gait rehabilitation because it enables the patients to identify their walking abnormalities and make corrections immediately.

In order to achieve intelligent gait analysis based on sensor signals, different sensor fusion and machine learning algorithms have been employed, such as fuzzy logic, hidden Markov models (HMM), support vector machine (SVM), and so on. Most of these algorithms, however, are computation-intensive. They require significant computational resources and cannot run in real-time during rehabilitation training. This poses a new challenge for the successful implementation of these algorithms for real-time gait analysis. On the other hand, the rapid development of big-data driven analytic techniques have been observed in many healthcare applications. It provides a new solution to fully enable the processing of massive sensory data for complex decision making and analysis. Motivated by this trend, in this chapter of this thesis, we propose a data-driven approach for real-time automatic gait phase detection, and implement it on a cloud-based gait monitoring and analysis platform. Our solution combines an infinite Gaussian mixture model
Signal processing unit
Battery
Bottom Side
Heel
Meta12
Meta45
Toe

Figure 3.1: An overview of the smart shoe design. A signal processing unit includes barometric sensors, microcontroller, and Bluetooth chip.

(IGMM) to classify different gait phases based on the GCF measurements, and a parallel particle filter to estimate and update model parameters. The parallel particle filter is abstracted as a computation topology and deployed on a parallel real-time computing framework in a Microsoft Azure cluster [3]. Both time-based and weight-based particle sharing mechanisms are also proposed to judiciously distribute particles among different working nodes and thus strike a good balance between computational overhead and estimation accuracy. The effectiveness of the proposed algorithm is validated using the data traces collected in a clinical study from five PD patients, three post-stroke patients, and three healthy subjects. Significant improvements on both computational efficiency and classification accuracy have been observed.

The remainder of this chapter is organized as follows. Section 3.1 introduces the infinite Gaussian mixture model. Section 3.2 presents the parallel particle filter approach and its particle sharing mechanisms. Section 3.3 describes the design and implementation of a cloud-based gait monitoring and analysis platform. The proposed algorithm is implemented and validated on this platform, and its performance is discussed in Section 3.4. Section 3.5 concludes the chapter.
3.1 Gait Phase Detection Using Infinite Gaussian Mixture Model

3.1.1 Smart Shoes for GCF Measurement

In order to better analyze patients’ gaits during walking, we have developed a pair of smart shoes to measure the GCFs on both feet [54, 122]. Fig. 3.1 gives an overview of the shoe design. Four barometric sensors are employed to measure the GCFs on the toe, the first and second metatarsophalangeal (MTP) joint (Meta12), the fourth and fifth metatarsophalangeal joint (Meta45), and the heel. Silicone tubes are wound into air bladders to connect barometric sensors with measurement ranging from 0 to 250 mbar. Each sensor can measure weight up to 200 lbs with a resolution of 0.2 lbs.

The pressure sensor outputs are read by a microcontroller through analog input channels and the sensor signals are sent out to a laptop or mobile device using a Bluetooth module. The Bluetooth module can reliably transmit signals in a range of 200 feet, which is enough for normal clinical and daily use. A 9-volt alkaline battery is used to power the smart shoes, and it can work consecutively for 90 minutes. The sampling rate of the smart shoes can go up to 100 Hz with the Bluetooth module. In our experiments the sampling rate is set at 20 Hz.

Although a healthy subject can have eight gait phases, it will not be possible to distinguish the three swing phases because the foot does not touch the ground in all these gait phases. As a result, the three swing phases are combined as one swing phase in this method and a maximum of six gait phases can be detected using only smart shoe measurement. However, many patients cannot fully release their toes, which cannot be observed easily by therapists but can be easily detected by the shoes. Despite different algorithms reviewed in the related work section for gait phase detection, they all require the users to input the number of gait phases to be detected, which makes them difficult and inaccurate to be employed for patients with various abnormal gait patterns.

Fig. 3.2 presents the representative raw data from a healthy subject, a PD patient and a post-
stroke patient, respectively. For the healthy subject, a gait cycle always starts with a strong heel strike, and then the subject moves the center of pressure to the forefoot before toe-off. Moreover, the subject is able to maintain good balance by allocating equal or more force to the medial boarder (Meta 12) in most of the gait cycles. However, for the PD patient, more force is observed on the lateral boarder (Meta 45) during the stance phase and this will significantly increase the risk of instability and falling. The stroke gait is even more abnormal, primarily due to the lack of heel strike as well as the poor stability shown by the large force on Meta 45. Additionally, the stroke patient walked much slower as it took 7 seconds to complete 3 steps, while the other two subjects completed 5 steps in less time.

3.1.2 Finite and Infinite Gaussian Mixture Models

The detection of gait phases based on the GCF measurements is essentially a classification problem. Our goal is to develop an algorithm that could automatically determine the number of existing gait phases for a patient. We will employ an Infinite Gaussian Mixture Model (IGMM) to achieve this goal, and it is an extension of a finite GMM, which is a powerful tool for classification and we assume the measurement data from the smart shoe comes from the following distribution

\[
p (y^{(i)}) = \sum_{k=1}^{K} p (c_i = k) p (y^{(i)} | \Theta_k),
\]  

(3.1)
where $K$ is the total number of gait phases to be detected, which needs to be predetermined by therapists. $y^{(i)}$ is the $i^{th}$ measurement data vector from the sensors (smart shoes in this case). $c_i$ indicates which gait phase the $i^{th}$ data vector belongs to, and $p(c_i = k) = \pi_k$ represents the a priori probability of the $i^{th}$ measurement data vector coming from the $k^{th}$ gait phase. $p(y^{(i)}|\Theta_k)$ follows the multivariate Gaussian distribution with the parameter $\Theta_k \sim (\mu_k, \Sigma_k)$ for each gait phase $k$. Maximum likelihood approach (such as expectation maximization) is typically used to estimate the model parameters $(\pi_k, \Theta_k)$.

Once the model parameters of the GMM have been identified, given a new measurement data vector $y^{(s)}$, the probability that it belongs to the $j^{th}$ gait phase is given by the following equation

$$p(c_s = j|\Pi, \Theta, y^{(s)}) = \frac{p(c_s = j) p(y^{(s)}|\Theta_j)}{\sum_{k=1}^{K} p(c_s = k) p(y^{(s)}|\Theta_k)}. \quad (3.2)$$

As is mentioned above, the total number of gait phases $K$ and the a priori probability need to be predetermined before applying GMM, which makes it difficult to apply in actual clinical environment. Moreover, it is not possible to detect gait phases using a single “best” model due to the complexity of human gait. In such case, it could be useful to report the gait phase detection as a result of an average of multiple models, which means the model parameters for the GMM are also random variables. One popular way of doing that is to use Bayesian approach as follows

$$p(M|y) \propto p(y|M) p(M), \quad (3.3)$$

where $p(M|y)$ is the posterior probability of a model $M$ given a set of measurement $y$, and $p(y|M)$ is the likelihood of observation $y$ given the model $M$. The choice of prior $p(M)$ is
given by \cite{117}

\[
\Pi | \alpha \sim \text{Dirichlet} \left( \frac{\alpha}{K}, \ldots, \frac{\alpha}{K} \right), \tag{3.4}
\]

\[
\Sigma_k \sim \text{inverseWishart}_{\nu_0} \left( \Lambda_0^{-1} \right), \tag{3.5}
\]

\[
\mu_k \sim \text{Gaussian} \left( \mu_0, \frac{\Sigma_k}{\gamma_0} \right), \tag{3.6}
\]

where $\alpha$ is the concentration parameter of the Dirichlet distribution. This Dirichlet distribution is used in this chapter to encode our prior knowledge on the number of gait phases and the possibility of each gait phase. The uniform parameterization in (3.4) is just for simplicity. However, some prior information about different gait phases can be incorporated such as the time allocation in gait phases in a healthy or pathological gait \cite{87}. The parameter set for this Bayesian mixture model (BMM) is written as $\chi \sim \left( \Lambda_0^{-1}, \nu_0, \mu_0, \gamma_0 \right)$. The joint distribution of the BMM is written as \cite{117}

\[
p \left( y, \Theta, C, \Pi, \alpha; \chi \right) = p \left( \alpha \right) p \left( \Pi | \alpha \right) \prod_{i=1}^{N} p \left( c_i \right) p \left( \Theta_i | c_i, \Pi \right) \prod_{j=1}^{K} p \left( \Theta_j ; \chi \right). \tag{3.7}
\]

Applying the Bayes rule (3.3) yields the following posterior probability conditioning on the observation data

\[
p \left( \Theta, C, \Pi, \alpha | y; \chi \right) \propto p \left( \alpha \right) p \left( \Pi | \alpha \right) \prod_{i=1}^{N} p \left( c_i | \Pi \right) \prod_{j=1}^{K} p \left( \Theta_j ; \chi \right). \tag{3.8}
\]

It is clear that the model (3.8) cannot be calculated analytically, so Markov Chain Mote Carlo (MCMC) method \cite{38} and the variational approach \cite{15} are used for calculating the posterior probability. The approach we employ in this chapter to deal with unknown number of gait phases $k$ is the IGMM model, which occurs as $K \rightarrow \infty$ for the BMM above. It means there could be
infinite choices of gait phases, which is a great representation of the complexity and variability of human gait. The limiting probability distribution function is given in [39] as:

\[ P(C|\alpha) = \alpha^{K_+} \left( \prod_{k=1}^{K_+} (m_k - 1)! \right) \frac{\Gamma(\alpha)}{\Gamma(N + \alpha)} \]  

(3.9)

where \( K_+ \) is the number of gait phases detected already, \( m_k \) is the number of data points in the \( k^{th} \) gait phase, and \( \Gamma() \) is the Gamma function. One popular way of sampling from the IGMM is by the following Gibbs sampler known as the Chinese restaurant process [9]

\[ p(c_i = k|C_{-i}) = \begin{cases} \frac{m_k}{i-1+\alpha}, & k \leq K_+ \\ \frac{\alpha}{i-1+\alpha}, & k > K_+ \end{cases} \]  

(3.10)

where \( c_i \) is the gait phase for the current data vector \( y^{(i)} \), and \( C_{-i} \) is the corresponding gait phases for all the previous data vectors. The first row of (3.10) indicates that the new data point belongs to one of the gait phases already detected, while the second means it belongs to a new gait phase. In the next section, we will develop a parallel particle filtering method to estimate the posterior probability and detect gait phases in real-time given a stream of measurement data vectors.

### 3.2 Parallel Particle Filter Design

Particle filter is a very popular method to solve non-linear estimation problems and has been used extensively in robotics and tracking-positioning applications [40], and more recently in other neural applications, like spike sorting [117]. In this section, we first give an overview of the particle filter method and then elaborate the design principles of the proposed parallel particle filter and its algorithm details.
3.2.1 An Overview of Particle Filter Method

In a particle filter, a set of $N$ weighted “particles”, $\{C^{(t)}_{1:N}, w^{(t)}_{1:N}\}$ (samples and associated weights at time $t$), are used to form a discrete representation of the distribution of interest ( posterior distribution over class identifiers given observations). In this chapter, our baseline solution employs the sequential particle filter method presented in [117]. The algorithm framework is summarized in Alg. 1 which comprises two main steps, sampling and resampling. The algorithm inputs are the data observations $y^{(1:T)}$ and the initial gait phases for the first $T_0$ datapoints. The algorithm then calculates the rest of the gait phases for datapoints $T_0 + 1$ to $T$ sequentially.

In the sampling phase a new diverse set of particles is generated by exhaustively enumerating all possible gait phases an observation $y^{(t)}$ belongs to. Initially, $M$ is calculated, which represents the total new particles generated (Line 2). Each particle, $C^{(t-1)}_{\{i\}}$ ($i \in [1, N]$), generates $K_{+(i)}^{(t)}$ (number of different gait phases detected so far at time $t$ for particle $i$) putative particles, $C^{'(t)}_{\{j\}}$, plus a new particle that represents a new gait phase being discovered (Line 5). These particles will take all the values in $[1, K_{+(i)}^{(t)} + 1]$. Each particle’s weight is updated (Line 7) by the new observation’s $y^{(t)}$ likelihood, using equations (14) and (16) described in [117] and multiplied by the prior probability (Line 6) of this gait phase using equation (3.10). At the end of the sampling phase all particle weights are normalized, so that their total sum is equal to 1 (Line 9).

In the resampling phase, the particles with negligible weights are replaced by new particles in the proximity of the particles with higher weights. As described in [33], the resampling step involves downsampling from $M$ particles to $N$. This is achieved by using the optimal resampling scheme that guarantees that there are no multiple copies of particles in the final set of $N$ particles. First, $c$ is calculated (Line 11) to be used in the threshold that compares with the weights of the particles. This threshold plays an important role in the downsampling step. Theorem 1 in [33] proves the correctness of the downsampling step. Particles that have weights greater than $\frac{1}{c}$ (those in set $S_1$, Line 12) do not need to be resampled. Otherwise, they will be resampled using the stratified resampling method [18] (Line 16), which is the most computationally efficient and
ensures that at most one copy of each particle is resampled. The probability of a particle being resampled is proportional to its weight. The stratified resampling method takes \( M - L \) putative particles as input, which is the cardinality of \( S_2 \) and returns only \( N - L \) particles. This ensures that the total number of particles in \( S_1 \) and \( S_2 \) is \( N \). Finally, the particles and their weights are updated to their best estimate up to the current observation (Line 17-18).

Algorithm 1 Particle filter for posterior estimation in IGMM

| Input: \( y^{(1:T)}, C^{(1:T_0)} \) |
| Output: \( C^{(1:T)}, w^{(1:T)} \) |

\begin{algorithm}
  \begin{algorithmic}
    \STATE for \( t = T_0 + 1 : T \) do
    \STATE \( M \leftarrow \sum_{i=1}^{N} K^{(t)}_{+_{(i)}} + N \)
    \STATE // Sampling phase
    \STATE for \( i = 1 : N \) do
    \STATE \( \text{generate particles } C^{(t)}_{(j)} \text{ with values from } 1 \) to \( K^{(t)}_{+_{(i)}} + 1 \)
    \STATE \( \text{calc. their prior } \pi_k \leftarrow P \left( C^{(t)}_{(j)} = k | C^{(1:t-1)}_{\{i\}} \right) \)
    \STATE \( \text{calc. } w^{(t)}_{(j)} \leftarrow P \left( y^{(t)} | C^{(1:t-1)}_{\{i\}}, C^{(t)}_{(j)}, y^{(1:t-1)}, ... \right) \times \pi_k \)
    \STATE end for
    \STATE normalize weights \( w^{(t)}_{\{m\}} \leftarrow w^{(t)}_{\{m\}} / \sum_{j=1}^{M} w^{(t)}_{\{j\}}, \forall m \)
    \STATE // Resampling phase
    \STATE \( c \leftarrow \text{such as } N = \sum_{j=1}^{M} \min(c \times w^{(t)}_{\{j\}}, 1) \)
    \STATE \( S_1 \leftarrow \{ C^{(t)}_{(j)} : w^{(t)}_{\{j\}} \geq \frac{1}{c}, \forall j \in [1, M] \} \)
    \STATE \( S_2 \leftarrow \{ C^{(t)}_{(j)} : w^{(t)}_{\{j\}} < \frac{1}{c}, \forall j \in [1, M] \} \)
    \STATE \text{cardinality of } S_1 \text{ is: } L \leftarrow \lvert S_1 \rvert \)
    \STATE assume \( W_1, W_2 \) the weights of \( S_1, S_2 \) respectively
    \STATE \( S_2 \leftarrow \text{stratified resample}(S_2, W_2, N - L) \)
    \STATE \( C^{(t)}_{\{1:N\}} \leftarrow S_1 \cup S_2 \)
    \STATE \( w^{(t)}_{\{1:N\}} \leftarrow W_1 \cup \{ \frac{1}{c}, \forall i \in [1, N - L] \} \)
  \end{algorithmic}
\end{algorithm}

3.2.2 Design Principles of the Parallel Particle Filter

The particle filter method presented in Section 3.2.1 is sequential and computation-intensive. Each sampling and resampling step involves a large number of numerical operations. By leveraging the cloud-based computing environment and distributing the computation workload among
multiple parallel working nodes, the execution time can be significantly reduced and thus enable real-time posterior estimation.

The main idea of our parallel particle filter is to divide the computation workload among multiple independent working nodes in the cloud computing environment, and perform parallel sampling and resampling. Essentially each working node implements a separate particle filter. The initial particles are distributed evenly across all the working nodes and each working node is responsible for maintaining, sampling and resampling its own particles. Only a lightweight central coordinator is needed to manage the weight normalization and particle sharing to ensure the quality of particles. By doing so, the computation workload on each node can be reduced and all nodes work collaboratively to finish the posterior estimation.

Many existing parallel filters resample sequentially or perform exactly the same resampling step in parallel which incur significant communication overhead. As can be observed in Alg. 1, updating the particles and their associated weights (Lines 17-18) is the main bottleneck of the parallel implementation. Except of the particle value, other related information like model parameters need to be copied or recalculated. The performance will degrade further when particles are distributed on different working nodes. Our parallel resampling does not perform the same resampling function as in the sequential particle filter method, but instead lets the nodes work individually on their local particles. This will greatly reduce the volume of exchanged messages. To maintain high estimation accuracy, effective particle sharing mechanisms are developed to move good particles among working nodes but only when necessary. This reduces the communication overhead between the working nodes and the central coordinator.

### 3.2.3 Particle Sharing Mechanisms

By changing the way particles are shared among the working nodes, it can affect both the execution time and estimation accuracy. The particle sharing mechanism involves moving particles from one working node to another, which introduces extra overheads. On the sender side, par-
Figure 3.3: An overview of the parallel particle filter
particles need to be identified, compressed and sent over the network to the destination; on the receiver side, the received particles need to be uncompressed and replace the local particles with smaller weights. Basically, the particle sharing mechanism executes Line [17] of Alg. [1] in a distributed fashion. Each working node calculates the total weight of its local particles. Nodes with larger total weights will share their particles to those nodes with smaller total weights of their local particles.

Particle sharing can either be time-based or event-based. In a time-based sharing mechanism, particles are shared at a fix number of rounds. Deciding the frequency of particle sharing however is challenging. On one hand, having particles shared frequently will introduce in large communication overhead and increase the total execution time. The posterior distribution estimation however will be more accurate as it performs similar to the sequential particle filter. On the other hand, having particles shared less frequently can decrease the communication overhead and each working node does not need to be busy waiting before they continue in the estimation given the next datapoint. However in this case, the algorithm may not give as accurate results as the sequential particle filter, because particles with lower weights will not be replaced, but resampled again. In addition to giving unstable and inaccurate results, this also poses the threat of increasing the total execution time. We have the observation in the experiments that when particles with small weights are not replaced but resampled repeatedly, they can enter a state of generating new classes, which can be explained from the Chinese restaurant process [117]. This will lead to increased sampling and resampling time.

To avoid these issues, we chose to design a weight-based particle sharing mechanism. The main idea is to keep the sum of particle weights on each working node higher than a threshold, and thus keep the estimation accuracy no lower than a certain level. The weight-based particle sharing mechanism is implemented on the central coordinator, and works in two phases. First, each working node sends the total weight of its local particles to the coordinator through a weight report. After the coordinator identifies any working node that has its local weight sum less than the threshold, it requests particles from the node with the highest total weight of local particles
by sending a share request. That working node will then send its particles to the coordinator through a share response. The coordinator, upon receiving the share response on the next round, will distribute the enclosed particles to all the requesting nodes. To reduce the communication overhead, the weight information and particle share request/response are encapsulated in one message to exchange between the working nodes and the coordinator.

Two key parameters in the weight-based sharing mechanism need to be carefully selected. The first one is the weight threshold to trigger particle sharing. It will significantly affect the algorithm performance in terms of both execution time and estimation accuracy. A low threshold value means more tolerance on “bad” particles, which in turn can have a bad impact on the estimation accuracy. Very low threshold could have an indirect effect on the execution time as well. Since particles with small weight are repeatedly resampled, this could increase the number of gait phase classes for those particles and thus increase the sampling/resampling time. A high threshold value can trigger more particle sharing and thus improve the estimation accuracy, while it also increases both the computation and communication overhead, giving a negative effect on the execution time. To decide the threshold value we first observe the normal value range of the sum of particle weights on working nodes. A good threshold value needs to be below the normal range so that the particle sharing is only triggered when the sum of a working node’s local particle weights drops below the normal expected values. Another key parameter is the number of working nodes involved in the computation. Generally, more working nodes lead to shorter execution time, as the particles are shared across more working nodes. This however could have an impact on the performance, since solving the particle filter with less particles may not make the result trustable. Generally, the number of particles that needs to be used in a particle filter depends on the quality of the data and the complexity of the distribution.
3.2.4 Algorithm Details

Fig. 3.3 gives a high-level overview of the data and control flow of the parallel particle filter. On the right side is the central coordinator with its execution sequence in each round. For simplicity, we only present the details of two working nodes. To show what happens with the particle sharing mechanism we illustrate the control flow at a “good” working node at the bottom left, i.e. it has the highest sum of local particle weights, and a “bad” working node at the top left, i.e. it has a sum of local particle weights lower than the threshold.

Alg. 2 summarizes the key steps each working node will follow to collectively implement the parallel particle filter. Notice that the number of particles involved in each working node here is $N' = \lceil \frac{N}{Q} \rceil$ and not $N$ as in the sequential particle filter, where $Q$ is the number of working nodes, i.e. the parallelism level. The algorithm defines a message structure, $sendMsg$ (Line 1), to send information to the central coordinator. This information includes the sum of weights (Line 10) for all the particles at the working node and possible particles that the working node has been requested to send (Line 25) to the coordinator. Lines 3 to 9 are similar to the sampling steps in the sequential algorithm 1.

The algorithm then waits for the response message, $recvMsg$, from the coordinator (Line 12). It includes the total particle weights from all the working nodes to be used for normalization (Line 14), as well as a possible share request (for good working node) or shared particles (for bad working node). In the case the received message contains particles to be shared with the current node (Line 16), the particles and their weights are stored for later use (Lines 30-31). Next the downsampling steps are executed, in a similar way as in the sequential algorithm. In Line 25, if the node receives a share request, it puts its best particles that do not need resampling in a share response message. Finally, in the resampling and particles construction steps (Lines 29-31) received particles (if any) will replace particles with small weight from $S_2$.

Alg. 3 summarizes the key steps of the central coordinator. They will be executed at each round (line 1) to serve the working nodes. The tasks of the coordinator are to calculate the sum
Algorithm 2 Working Node in Parallel Particle Filter

1: define message $sendMsg \leftarrow \{}$
2: for $t = T_0 + 1 : T$ do
3:  $M \leftarrow \sum_{i=1}^{N} K_{+}^{(t)} + N$
4: // Sampling phase
5: for $i = 1 : N$ do
6:  generate particles $C_{\{j\}}^{(t)}$ with values from $1$ to $K_{+}^{(t)} + 1$
7:  calc. their prior $\pi_{k} \leftarrow P(C_{\{j\}}^{(t)} = k | C_{\{i\}}^{(t-1)})$
8:  calc. $w_{\{j\}}^{(t)} \leftarrow P(y^{(t)} | C_{\{i\}}^{(1:t-1)}, C_{\{j\}}^{(t)}, y^{(1:t-1)}, ..) \times \pi_{k}$
9: end for
10: $sendMsg \cdot weight \leftarrow \sum_{j=1}^{M} w_{\{j\}}^{(t)}$
11: $send\_to\_coordinator(sendMsg)$
12: $recv\_from\_coordinator(recvMsg)$
13: // Resampling phase
14: normalize weight $w_{\{m\}}^{(t)} \leftarrow w_{\{m\}}^{(t)}/recvMsg.\_wSum, \forall m$
15: $S_{recv} \leftarrow \{\}; W_{recv} \leftarrow \{\}; K \leftarrow 0$
16: if $recvMsg$ contains particles then
17:  $S_{recv} \leftarrow recvMsg.\_particles$
18:  $W_{recv} \leftarrow$ corresponding weights of $S_{recv}$
19:  cardinality: $K \leftarrow |S_{recv}|$
20: end if
21: calculate $c$, such as $N' = \sum_{j=1}^{M} \min(c \times w_{\{j\}}^{(t)}, 1)$
22: $S_1 \leftarrow \{C_{\{j\}}^{(t)} : w_{\{j\}}^{(t)} \geq \frac{1}{c}, \forall j \in [1, M]\}$
23: $S_2 \leftarrow \{C_{\{j\}}^{(t)} : w_{\{j\}}^{(t)} < \frac{1}{c}, \forall j \in [1, M]\}$
24: if $recvMsg$ contains share request then
25:  $sendMsg.\_particles \leftarrow S_1$
26: end if
27: cardinality of $S_1$ is: $L \leftarrow |S_1|$
28: assume $W_1, W_2$ the weights for $S_1, S_2$ respectively
29: $S_2 \leftarrow stratified\_resample(S_2, W_2, N' - L - K)$
30: $C_{\{1:N'\}}^{(t)} \leftarrow S_1 \cup S_2 \cup S_{recv}$
31: $w_{\{1:N'\}}^{(t)} \leftarrow W_1 \cup \{\frac{1}{c}, \forall i \in [1, N' - L - K]\} \cup W_{recv}$
32: end for
Algorithm 3 Central Coordinator in Parallel Particle Filter

1: while TRUE do
2:   particles ← ∅
3:   for all working nodes $q = 1 : Q$ do
4:     recv($Msg(q)$)
5:     if $Msg(q)$ contains particles then
6:       particles ← $Msg(q).particles$
7:     end if
8:   end for
9:   find $\hat{Q}$, the working node with highest sum of weights
10:  totalWeight ← $\sum_{q=1}^{Q} Msg(q).weight$
11:  for all working nodes $q = 1 : Q$ do
12:     if $Msg(q).weight < weightThreshold$ then
13:       if particles $\neq \emptyset$ then
14:         $Msg(q).particles$ ← particles
15:     else
16:         $Msg(\hat{Q}).shareRequest$ ← TRUE
17:     end if
18:   end if
19:  $Msg(q).weight$ ← totalWeight
20: end for
21: $\forall q \in [1, Q] : send(Msg(q), q)$
22: end while
of the particle weights across all working nodes, and to perform particle sharing if necessary. The coordinator first starts with receiving messages from all the working nodes (Line 4). If the received message contains particles to be shared, the coordinator stores them in memory for later use (Line 6). After receiving messages from all the working nodes, the coordinator finds the working node \( \hat{Q} \) that has the highest sum of local particle weights and calculates the total particles weight (Lines 9-10). In the next step, for each working node the coordinator checks if its total local weight is less than the threshold value. If the bad working nodes are found and in the current round particles have been received from a good the working node, these particles are shared to the bad nodes (Line 14), otherwise a share request is sent to \( \hat{Q} \) (Line 16). In the next round \( \hat{Q} \) will share its best particles, i.e. those particles in its \( S_1 \) set (Alg. 1, Line 25), so that the coordinator can share them with any node that had weight lower than the threshold value. Finally in Lines 19 and 21 the total weight is updated in the message and sent to every working node.

### 3.3 System Implementation

We developed a cloud-based gait monitoring and analysis platform to validate our algorithm design. The platform comprises a front-end sensing system (smart shoes) to collect the GCF measurements, and a real-time computing framework on Microsoft Azure to perform real-time gait phase detection and analysis. The analytics platform can either run on a private computing infrastructure or deployed on an enterprise cloud platform. The proposed system architecture is presented in Fig. 3.4 which follows a Client/Server architecture design. The server provides high-volume data ingest, scalable time-series data storage and real-time parallel data processing. The clients can either push real-time sensor data streams into the server or acquire data (in the formats of query results or graphic visualization) from the server.

We created a TCP server running on Netty as the portal virtual machine (VM) to accept the meta/raw sensory data from external data sources (both patients and therapists) using a unified JSON format. A combination of Apache Kafka [5] and Storm [6] frameworks is running in an
HDInsight cluster for real-time delivery and processing on data received from the TCP server. Raw sensory data are also sent to an HDInsight HBase [4] cluster for offline batch processing when necessary. Power BI and dashboards were used for reporting, real-time alerts and notification. A model editor is developed as well to specify the computation topology which will be deployed on Apache Storm for designated real-time data processing.

We choose Apache Storm to be the real-time processing framework because Storm makes it easy to reliably process unbounded data streams. It uses custom created “spouts” and “bolts” to define information sources and manipulations to allow batch, distributed processing of streaming data. A Storm application is designed as a “topology” in the shape of a directed acyclic graph
In our Storm implementation (see Fig. 3.5), the data are collected from the smart shoes and linked to the spout through the TCP server. The spout then passes the data to the computation nodes, which are implemented in working bolts. Each working bolt executes the working node program (Alg. 2). By varying the parallelism parameter of the topology, we can define the number of nodes that work collaboratively for gait phase detection. The central coordinator program (Alg. 3) is implemented in a central coordinator bolt, and data from all working bolts are shared to it. In the coordinator bolt the particle weights are summed up and returned to the working bolts to continue the execution. The coordinator bolt also runs the particle sharing mechanism, to check if the local total weight for every working bolt drops below the threshold value. If so, the node with the highest local total weight is requested to share its particles with the nodes with small weights on the next round. Messages in the format of tuples are sent both
from the spout to the working bolts and between the working/coordinator bolts. The output of this Storm topology is a collection of all particles with their corresponding weights. It can be collected in an output bolt and subscribed by any consumer.

3.4 Performance Evaluation

In this section, we present three sets of experiments for the performance evaluation on the proposed gait phase detection algorithms. We first describe the experimental settings and the dataset used for the evaluation. The first set of experiments evaluate the estimation accuracy of the gait phase detection algorithms on healthy subjects, PD patients and post-stroke patients. The second set of experiments study the computational efficiency of the algorithm implementation on the cloud-based gait monitoring and analysis platform. The third set of experiments investigate the trade-off between computational efficiency and estimation accuracy of the algorithms with varied number of particles.

3.4.1 Experimental Settings

We have implemented the gait phase detection algorithms on our platform consisting of 3 VMware servers. Each server is equipped with 16 Intel Xeon cores, clocked at 2.10GHz and 64 GB of memory. Four virtual machines (VM) are installed on each server, running Ubuntu Linux 14.04 LTS. Each VM is configured to use 4 single core processors and 12 GB of memory. On these VMs installed are: 1 nimbus node, 3 zookeeper nodes and 11 supervisors. Each supervisor has four slots (Java Virtual Machines).

In order to evaluate the performance of the proposed algorithms, we collected GCF data, using the developed smart shoes, from both healthy subjects without known walking problems and PD and post-stroke patients. Experiments with healthy subjects were conducted in the Mechanical Systems Control Laboratory at the University of California, Berkeley. The clinical study
with PD and post-stroke patients was conducted in the William J. Rutter Center at the University of California, San Francisco (UCSF). The Committee on Human Research (CHR) at UCSF reviewed and approved this study. The original purpose of this human subject study was to examine whether patients could use visual feedback to direct their rehabilitation training and how was the training performance compared to traditional rehabilitation training directed by a physical therapist only. We use these datasets to evaluate the algorithms developed in this chapter. Detailed experimental design and statistical analysis of the clinical outcomes are available in [17, 122].

To collect data for this work, the subjects were asked to walk multiple trials on a flat ground for at least 50 consecutive steps in their normal walking speeds. The data collected from five PD patients, three post-stroke patients, and three healthy subjects are used to test our methodology. The average ages for each group are 69.2, 53 and 23 years old respectively. In total, 403 data traces were extracted from the collected data.

In the first two sets of experiments, we used a same number of 450 particles in both sequential and parallel particle filter implementations for fair comparison. In the last set of experiments, the number of particles was varied to study its effect on the estimation accuracy and computational efficiency of the algorithms. Furthermore, in all the experiments, 30% of the data points were selected to initialize the first $T_0$ data points. The gait phases for those observations were calculated using Gibbs sampling, which is a Markov Chain Monte Carlo (MCMC) algorithm, but does not support real-time gait phase detection.

### 3.4.2 Estimation accuracy of the gait phase detection algorithms

To evaluate the estimation accuracy of the gait phase detection algorithms, we run the algorithms for each of the 403 traces. Due to the space limit, only one trace is shown for each of the three subject classes, i.e., healthy subject (Fig. 3.6), PD patient (Fig. 3.7) and post-stroke patient (Fig. 3.8). Out of the 450 total particles that were used to represent the posterior distribution, we only present gait phases from the one with the highest weight. The raw GCF measurement
from one shoe is plotted in sub-figure (a) of each figure. The gait phases detected from the particle filter algorithm can be seen in sub-figure (b) of each figure. For each subject class, the algorithm detected four distinct gait phases. Gait phase 0, 1, 2, and 3 refers to the swing, initial contact, mid stance, and terminal stance phase, respectively. Please note here that in the healthy subject’s gait phases (Fig. 3.6b), gait phase 1 (initial contact) is only detected instantaneously for one observation, making the plot look like this phase is never detected. In sub-figure (c) of each figure, gait phases detected from fuzzy logic are displayed [54]. It is assumed that there are six gait phases, with labels Swing, IC, LS, MS, TS and PS referring to swing, initial contact, loading response, mid stance, terminal stance, and pre-swing phases, respectively.

Figure 3.6: (a) Raw GCF measurements from a healthy subject, (b) gait phases detected from the particle filter algorithm, (c) gait phases detected from fuzzy logic.

The main advantage of the gait phase detection algorithm based on particle filter is that the physical therapist who is going to use the algorithm does not need to manually configure the
Figure 3.7: (a) Raw GCF measurements from a PD patient, (b) gait phases detected from the particle filter algorithm, (c) gait phases detected from fuzzy logic.
Figure 3.8: (a) Raw GCF measurements from a post-stroke patient, (b) gait phases detected from the particle filter algorithm, (c) gait phases detected from fuzzy logic.
parameters for the algorithm, except 1) the model priors which are done only once, and 2) the number of particles that are going to be used. On the other side, the fuzzy logic approach requires users to input the number of gait phases and the threshold values for each gait phase, which need to be tuned for each subject. This is challenging if not impossible, as the number of gait phases is not always known a priori, and tuning the threshold values are also subjective and time consuming.

By observing the output from the particle filter based gait phase detection approach, we also can observe that the algorithm only detected 4 gait phases in total. That happened mainly for two reasons: the selection of model priors and the particle initialization. It is important to note here that this does not indicate that our gait phase detection results are wrong, as in rehabilitative training, it is more important to know if the algorithm can detect only the specific gait phases that are related to the gait abnormality of the subject. It also needs to be clarified that there is no single correct answer of gait phase detection due to the complex human walking dynamics, so we used the fuzzy logic approach as a cross-validation, but not as a ground truth. Further research towards this direction is needed in cooperation with physical therapists, so meaningful metrics from the extracted gait phases can be developed.

Similarly, by observing the output from the fuzzy logic approach we can see that although 6 gait phases were asked to be found, only 4 phases were found for the healthy subject and the post-stroke patient, and 5 phases for the PD patient. As pointed out before, precise threshold values for the fuzzy logic approach are very important for the detection of gait phases. Nonetheless, the fact that both algorithms identified less than 6 gait phases, indicates that 6 gait phases cannot be expressed from these selected data traces.

By comparing the output from both algorithms we observe similar behaviors of the detected gait phases from the healthy subject but some differences in the detected gait phases from the PD and post-stroke patients’ data traces. For the healthy subject’s data, both algorithms detect a very short initial contact, followed by a longer med stance and a short terminal stance. On the other hand, the fuzzy logic approach detected a significantly longer initial contact for both patients’
data, followed by a short loading response, skipping mid stance and ending with terminal stance and pre-swing phases. The absence of the mid-stance phase could be caused by the reduced force to the medial boarder (Meta 12) and increased force on the lateral boarder (Meta 45) during the stance phase, which can significantly increase the risk of instability and falling. Moreover, the increased length of the pre-swing phase can be explained by the large force on Meta 45 and toe. One advantage of the particle filter algorithm can be seen here is that, regardless of the force levels of the sensor data, it can still find a short initial contact, followed by a longer stance phase and ending a gait cycle with a terminal stance phase. This can be attributed to the non-parametric nature of IGMM.

Finally, we observe that the particle filter based algorithm returns some unexpected changes or jumps in the gait phases. One is between 2.5 and 3 seconds in the data trace of the healthy human subject and the other one is between 7 and 7.5 seconds in the data trace of the PD patient. The jump observed from the PD patient’s gait is easier to explain as data from gait phases 2 and 3 are close and it could be the case that the probability of having gait phase 2 instead of 3 is higher given the observation originating from the four sensors at the specific time point. However, the jump observed in the healthy subject’s data is not consistent with how data points from phase 0 (swing phase) should look like and needs further investigation. We believe the probabilistic nature of the IGMM algorithm is the reason for this unexpected jump. Further research can be done to explain why this happens and filter such unexpected changes in the data with additional smoothing layers in the prediction of the final gait phase to improve the robustness of the algorithm. Nevertheless, it is clear that the particle filter based algorithm gives accurate gait phase detection results without subjective human input which requires significant clinical experience.
3.4.3 Computational efficiency of the proposed methods

To evaluate the computational efficiency of the implementation with varied levels of parallelism, we conducted a set of experiments with different number of workers running the Storm topology. Fig. 3.9 summarizes the execution times of the parallel particle filter method tested on the Storm cluster, running for the same dataset presented in Section 3.4.2. The number of Storm workers we have tested in the experiments is varied from 6 to 26. In each experiment, the particles were evenly distributed to all working nodes, so that each working node had an equal amount of particles for sampling and resampling. Since particle filters are sampling methods, the execution time can vary. This also depends on the number of detected classes per particle. If the total number of detected classes is large, the sampling and resampling tasks can take more time to complete. To cope with this issue, we repeated the experiments 20 times and present the mean execution times. From Fig. 3.9 we have the observation that the shortest average execution time is a little over 4 seconds which happened when 14 working nodes were used, or more. Additionally, the only notable difference between the execution times for each subject class is that the healthy subject’s total execution time (Fig. 3.9a) is slightly shorter than the other two. This can be explained because the length of the input for the healthy subject is around 1 second shorter than the two patient’s data length.

With the same experiment settings, the sequential particle filter approach took 37.7 seconds on average to process the data traces from the healthy subject and the PD patient, while for the post-stroke patient’s data trace it took 37.1 seconds on average. From the comparison, it is clear that the parallel particle filter approach can significantly improve the execution time, making it capable of solving the gait phase detection problem in real-time. Another interesting observation in Fig. 3.9 is that after the parallelism level increases beyond 14, the improvement on execution time is not significant.

Sampling, resampling, and network message exchange are the three major contributors to the total execution time, and their time distributions are also summarized in Fig. 3.9. All the
Figure 3.9: The distribution of the execution time of the gait phase detection algorithm running with varied workers for different patient classes.
results shown in this figure are averaged among 20 different runs. The network latency is the total average latency across all the messages sent from the working nodes to the central node and back. From Fig. 3.9, we have the observation that sampling is the most time consuming task in the particle filter approach, and distributing the workload (by distributing the particles across different workers) helps reduce the sampling time. The second most time consuming task is resampling, which is the main reason why our implementation employs parallel instead of sequential resampling. We also can observe that along with the increase of the number of working nodes, both sampling and resampling time decreased. Moreover, we observe that the network latency is slightly increased as the number of working nodes is increased and when 22 or 26 working nodes are used it is slightly larger than the re-sampling time. This is mainly due to the fact that the nodes having particles with higher weights may finish faster than others, but still have to wait for the slower working nodes to send their messages to the central coordinator and hear back. All this busy waiting time is reflected as network latency in Fig. 3.9. Finally, it needs to be noted that the summation of sampling, resampling and network latency does not precisely add up to the total execution time. This is because the times reported here are the average values across 20 different runs. The total execution time in each round is determined by the slowest node, as all the other nodes need to wait for it to finish. Since every other node spent less time in sampling and resampling, the summation of the average sampling, resampling and network latency times, per round, will not add up to the total execution time. The closer is this summation to the total execution time, the more significant speedup can be achieved by the algorithm and the less number of workers are blocked by slow worker(s).

3.4.4 Determining number of particles

To run the particle filter based algorithm, the number of particles used for the estimation needs to be specified. Generally, the more particles used, the better estimation of the posterior distribution will be. However, with more particles used, the particle filter algorithm will require
more computing resources to achieve the real-time constraints. Thus, it is challenging to select the appropriate number of particles that will effectively balance the trade-off between estimation accuracy and real-time constraints. To study the effect of the selected particle number on the estimation accuracy and execution time, we designed the third set of experiments. In this set of experiments, we ran the particle filter algorithm on the data trace collected from a healthy subject (Fig. 3.6a) with the number of workers fixed to 6 and the particles number varied from 20 to 200 with an increment of 20 and from 200 to 700 with an increment of 50. To report the accuracy of the estimation, the performance metric of fault rate was used, which measures the percentage of observed wrong gait phase changes. Within a gait cycle, any gait phase needs to be followed by the next phase in the sequence (Fig. 2.1). Any gait phase change that does not follow this sequence is considered wrong (at least for healthy gait). The results of this set of experiments are summarized in Fig. 3.10. Execution times are also reported for the corresponding number of particles selected. The results are averaged across 20 different runs to account for possible unexpected behavior.

From Fig. 3.10 we have the following observations. First of all, the execution time of the gait phase detection algorithm is proportional to the number of used particles. When more than 550 particles were used, the execution time exceeded 9 seconds, which was the data input length. Another observation is that the fault rate was around 0.86 when 20 particles were used but de-
increased significantly to less than 0.27 as the number of used particles increased from 40 up to 700, with the smallest value of 0.17. A sample output of the algorithm with 20 particles can be found in Fig. 3.11, which was not robust. When new observations were processed, new gait phases were discovered and there does not seem to be any clear pattern in the new phases. Only phase 0 (the swing phase) seems to follow the periodic cycle. This indicates that more particles indeed help provide more collective memory, which in turn can eliminate unstable behaviours.

Please note that for the data trace shown in Fig. 3.10 which is collected from a healthy human subject, selecting more than 40 particles for gait phase detection seems sufficient. We chose 450 particles in the first two set of experiments to deal with the uncertainty of pathological gait.

### 3.5 Conclusion

The evergrowing demand on gait rehabilitation requires the gait analysis to be objective, automatic and scalable. To address these challenges, the method that was presented in this chapter
presents a data-driven approach for real-time gait phase detection to facilitate gait analysis and rehabilitation. The approach combines an infinite Gaussian mixture model (IGMM) and a parallel particle filter to classify the gait phases and update the model parameters. Both time- and weight-based particle sharing mechanisms are designed to distribute particles among different working nodes to strike a good balance between estimation accuracy and computational overhead. The proposed gait phase detection algorithm is implemented on our gait monitoring and analysis platform developed on our VMware server and validated using the dataset collected in clinical study with five PD patients, three post-stroke patients, and three healthy subjects.

This gait phase detection algorithm is used as a fundamental component of the whole algorithmic framework for gait analysis and biometric authentication that is presented in this thesis. Once gait phases have been objectively detected with this algorithm, more sensing features related to mobility, balance, strength can be extracted within the extracted gait phases and gait cycles. In addition, we extract meta-features based on the identified gait phases from this algorithm. All these features can be used in gait monitoring during rehabilitative training, or can be used as features in objective gait disorder diagnosis. In the next chapter of this thesis we present a novel framework for diagnosis of gait abnormalities that can be attributed to different underlying neurological disorders.
Chapter 4

Classification of Neurological Gait Disorders Using Multi-task Feature Learning

In the current practice, gait rehabilitative therapy is provided by therapists who manually stimulate patients’ reflexes and rotate their lower limbs to retrain their central nervous system with the correct gait patterns. This approach is not only physically demanding for both patients and therapists, but also expensive and time-consuming. Moreover, in the clinic, assessment of gait abnormalities is based on timed tests, visual observations by therapists, retrospective qualitative evaluations of video tapes, and specific physical tests, e.g., strength, range of motion, balance, gait speed, and endurance. As a result, most times gait assessment is based on the subjective judgment of the therapist. More objective methods are desired to quantify the gait assessment and progress evaluation of the rehabilitative training, reduce the chances of biased assessment by therapists, and provide better, targeted treatment to patients. To further improve and help in the automation of diagnostic practices in gait analysis, in this chapter we present an integrative framework to automatically classify gait disorders from two common neurological diseases, stroke and PD, and distinguish abnormal gait caused by these two diseases from the healthy gait.
Because there is strong correlation between the two neurological diseases and resultant gait disorders, multi-task machine learning strategies can be more feasible to identify similarities and differences of gait patterns than classic multi-class classification algorithms given the latter focus on modeling only the exclusive (or discriminative) features of different gait classes \cite{76, 114}. An advanced multi-task learning algorithm has been developed to jointly create three classifiers, respectively, for distinguishing stroke-induced gait from healthy gait, PD-induced gait from healthy gait, and PD-induced gait from stroke-induced gait. To evaluate the proposed methodology we use data from a human participant study, which includes five PD patients, three post-stroke patients, and three healthy subjects. In our experiments the classification performance achieved Area Under the Curve (AUC) score of at least 0.96. The advantage of our multi-task learning method is that it can identify features useful for all three classification tasks as well as those predictive of a specific abnormality. We conclude our evaluation with a discussion on the important sensing features identified by the algorithms.

The remainder of the chapter is organized as follows. Section 4.1 discusses the gait sensing features we extracted based on the data. In Section 4.2, we introduce the multi-task learning approach and use it to classify gait based on the extracted sensing features. Evaluation results are given based on the recorded data from a human participant study and findings are summarized in Section 4.3. We conclude this chapter in Section 4.4.

### 4.1 Gait Features Extraction

To accurately describe specific human gait disorders is often a difficult task \cite{99}. Consequently, it is challenging to devise gait sensing features\footnote{In the remainder of this chapter we refer to gait parameters, the term used in most literature studies, as gait features to avoid confusion with the model parameters used in the multi-task learning methods in Section 4.2} that can be used to classify gait patterns. Furthermore, the GCF data collected from the smart shoes (Section 3.1.1) can be noisy due to imperfect sensor dynamics and complexity of human gait. In this section we present a set of gait fea-
In Table 4.1, fourteen gait features are proposed based on the GCF data collected from the smart shoes. These features are organized into four categories: gait phases, mobility, balance and strength. Their details will be discussed in the following subsections. Among these features, double support ratio, single support ratio and cadence are comprehensive features, which require bilateral information. All the other features are unilateral, as they can be calculated for each side separately [113]. Gait phase features are based on our previous work [80] and all the others are inspired by [108].

4.1.1 Gait Cycles

We first give an overview of what a gait cycle is, as all the gait features are extracted once for each gait cycle in a walking trial. Gait cycle is the time interval between the same repetitive events of walking. The defined cycle can start at any moment, but generally begins when one foot contacts the ground. If it starts with the right foot contacting the ground, the cycle ends when the right foot makes contact again. Fig. 4.1 gives an overview of two gait cycles at the lower two
Figure 4.1: An overview of a gait cycle and the gait features from four categories: gait phases, mobility, balance and strength.

The gait cycle can be broadly divided into two phases: stance phase and swing phase \[28\]. These two phases can then be further divided into sub-phases within the gait cycle, as shown at the top part of Fig. 4.1. In general, the stance phase takes around 60\% of the gait cycle \[28\] and can be divided into double support and single support. In double support, both feet are in contact with the ground, while in single support only one foot is in contact with the ground. Double or single support ratio refers to the portion of time within a gait cycle someone spends in double or single support respectively. The swing phase is described when the limb is not weight bearing and represents around 40\% of a single gait cycle \[28\]. These percentages can change with the walking speed, as with a higher speed the double support ratio in the gait cycle tends to be reduced. In Fig. 4.1 the lower depicted cycle starts with right foot initial contact, which leads to the stance phase, while the other starts with left pre-swing phase which leads to swing phase. Indicative percentages are shown to indicate the different phases within the cycle.

In Fig. 4.1 different gait features are shown for different categories, like mobility, balance, strength. Gait phases are shown at the top of the figure. In Sections 4.1.2 and 4.1.3 we discuss how gait phases are extracted and what gait phase related features are used in this work for gait
disorder diagnosis. In Sections 4.1.4 and 4.1.5 we discuss other features related to mobility, balance and strength.

4.1.2 Gait Phase Detection

Gait phases refer to various states within one walking cycle, and there are typically eight gait phases for a healthy subject (as shown at the top of Fig. 4.1): initial contact, loading response (or pre-swing), mid-stance, terminal stance (or initial contact), pre-swing, initial swing (not shown in Fig. 4.1), mid-swing, and terminal swing [54, 122]. Pathological gait can be unpredictable and complex, thus some gait phases might be missing and the time allocation of gait phases might also be different from a normal gait. This abnormal gait phase allocation provides a powerful tool for abnormal gait detection.

In this work we extract gait phase related features based on our previous work which applies infinite Gaussian mixture modeling, a non-parametric Bayesian method, for gait phase detection [80]. Our approach estimates the unknown number of gait phases that can be best described from the GCF data. Particle filters and the popular chinese restaurant process (CRP) are used for online model parameters estimation. In the rest of this subsection we describe how swing and stance phases are identified from the extracted gait phases.

Identifying swing phases from the unlabeled gait phases is important as many other gait features are based on it. Although it is straightforward to find healthy gait’s swing phase (Fig. 4.1), the swing phase detection in pathological gait can be challenging for multiple reasons. First of all, the way smart shoes are worn can affect the raw GCF sensor signals. Tight shoe laces will change the raw values recorded by the barometric sensor, leading to different absolute values even for the same person in different sessions. Additionally, the stochastic nature of the sampling, which is used to estimate the distribution of gait phases [80], can sometimes introduce new gait phases, which are not eventually represented in the GCF data. Finally, pathological gait can be so complex that sometimes new gait phases are explored from the particle filter algorithm.
Apart from that, various conditions of neural or muscular impairments, like foot-drop, can cause fore-foot dragging on the ground [28]. In such cases new gait phases are likely to be discovered and they should be identified as swing phases. Having correctly identified swing phases is very important as many other features are based on them.

As discussed earlier, the swing phase ratio (portion of time spent on swing phase) of a healthy gait is typically around 40% of the gait cycle [28]. This may change depending on the walking speed. Pathological gait can have smaller swing phase ratio, as the patient is walking slowly. Also, in the swing phase, GCF measurements will take very small positive values (or zero), as pressure from the body is not present in that limb. Using these two properties we identify the swing phases from all discovered gait phases according to the following steps:

We first calculate the average euclidean distance for all the observations in each gait phase from 0, by taking its 2-norm. We then sort the gait phases in increasing order based on their norms. We create a new swing phase, and add the observations in the sorted gait phase list one by one until the total number of observations in the new swing phase is more than 10% of all the observations. The 10% threshold is empirically chosen and gives the desired swing phase ratio in our dataset. The number of swing phases that were merged is kept as it is used as a gait phase feature (see Sec. 4.1.3). All the extracted gait phase features are described in the following subsection.

4.1.3 Gait Phase Features

The gait phase features are calculated from the gait phases that are extracted by our gait phase detection algorithm (see Section 4.1.2 and for more details, please refer to [80]). The expected number of gait phases can be calculated from the particles and their weights returned from the particle filter algorithm as \( \bar{K} = \sum_{i=1}^{N} w_i K_i \), where \( K_i \) is the number of gait phases detected from particle \( i \) and \( w_i \) is the particle’s weight. \( \bar{K} \) is a measure of complexity of the human gait. Compared with the eight standard gait phases of a healthy subject, pathological gait is unpredictable.
and it may have a different number of gait phases. For example, post-stroke patients with affected neurological system may experience foot-drop. This usually increases the stance phase with circumduction to allow toe clearance [28], which can lead to toe dragging on the ground, and thus causing the gait phase detection algorithm detecting multiple swing phases. The number of swing phases is another gait parameter and has been discussed in the previous Section 4.1.2.

The symmetry of gait phases (swing phases) is used as a measure to quantify how even the proportion of time spent is in each gait phase in a gait cycle (swing phases). We chose to include this new type of symmetry measure as it can be easily applied on the gait phases that were extracted from our Dirichlet process mixture model [80], given the fact that the number of gait phases is not known \textit{a-priori} for each subject. This single gait parameter can estimate the symmetry for any number of gait phases detected. It is based on the cosine similarity, as described in the following formula:

\[
\cos(\theta) = \frac{\mathbf{g} \cdot \mathbf{u}^T}{||\mathbf{g}|| \cdot ||\mathbf{u}||} \tag{4.1}
\]

where \(\theta\) is the angle between \(\mathbf{g}\) and \(\mathbf{u}\), with \(\mathbf{g}, \mathbf{u} \in \mathbb{N}^K\) and \(K\) is the number of gait phases (swing phases) found. \(\mathbf{g}\) is a vector of size \(K\), where each element in \(\mathbf{g}\) counts the number of observations belonging to each gait phase (swing phase) within a gait cycle and \(\mathbf{u}\) is a vector of size \(K\) with all its elements equal to 1, assuming observations in \(\mathbf{u}\) are evenly distributed. If the number of observations belonging to each gait phase is not evenly distributed and thus there are gait phases with very few observations, the angle between vector \(\mathbf{g}\) and \(\mathbf{u}\) will be higher resulting in lower symmetry. On the other hand, if the number of observations belonging to one gait phase is always more than normal it would also result in lower symmetry.

4.1.4 Mobility Features

Bradykinesia (slow walking) is one of the many characteristics of both stroke and PD gait [28, 118]. To capture similar characteristics we select four features in the mobility category, cadence,
double and single support ratios and stance phase ratio. Bradykinin can lead to increased double support ratio and cadence. Cadence is measured in steps per minute and it is calculated by taking the total number of stance phases in one trial divided by the length of the trial in minutes. The double support ratio refers to the proportion of time in a gait cycle that both feet are in the stance phase to support the subject, whereas the single support ratio refers to the proportion of time in a gait cycle that only one foot touches the ground while the other is in the swing phase. Stance phase ratio refers to the proportion of time in a gait cycle that one foot is in the stance phase. All these features are summarized in Fig. 4.1.

4.1.5 Balance and Strength Features

Patients with neurological related diseases, like stroke and PD, may experience weak muscle strength and balance [28]. We select two features in the balance and strength categories each. In the balance category, the maximum and minimum force differences between the medial (Meta12, Fig. 3.1) and lateral (Meta45, Fig. 3.1) sides of the forefoot in a gait cycle can be calculated as

\[
\begin{align*}
\max_{i \in I} F_{M12}(i) - F_{M45}(i), \\
\min_{i \in I} F_{M12}(i) - F_{M45}(i).
\end{align*}
\]

These features can evaluate the capability of maintaining balance. The \( I \) refers to the set of indices \( i \) that belong to one gait cycle. Strength is quantified using the maximum force on the heel during heel strike and on the toe during toe off. All balance and strength features are normalized by the body weight to make them comparable among different subjects.

4.2 Multi-Task Feature Learning for Gait Disorder Diagnosis

Based on the extracted gait features, we diagnose gait disorders by constructing classifiers as functions of these features. In this work, we use an advanced multi-task feature learning (MTFL)
classification method [114] to build three classifiers to discriminate gait observations of PD and stroke patients, respectively, from those of healthy adults as well as in between the gaits of PD and stroke patients. The selected learning strategies can possibly identify similarities and differences of gait patterns than classic multi-class classification algorithms given multi-class classification methods focus on modeling only the exclusive (or discriminative) features of the different gait classes. Moreover, the methodology helps in important gait feature selection which may help in better understanding the key characteristics that distinguish abnormal gaits and help design more targeted treatment methods.

MTL is a methodology that can improve the generalization of multiple related classification tasks by exploiting the task relationships, especially when the training set for some or all the tasks is limited. Related tasks are learned in a joint manner, so that knowledge learned from one task may benefit learning for other tasks. For example, in gait disorder diagnosis, the task of deciding if an observation, represented by a vector of gait features, is recorded from a PD patient or healthy subject, may help diagnose if another observation is recorded from a post-stroke patient or a healthy subject. MTL has been shown to be theoretically and practically more effective than learning tasks individually [114]. A widely-used basic assumption is that the related tasks may share a common representation in the feature space, which is investigated by multi-task feature learning (MTFL).

We revisit two of our recently developed MTFL methods that both rely on a multiplicative decomposition of the model parameters used for each task, and hence are referred to as Multiplicative MTFL (MMTFL). Both methods are related to the widely used block-wise joint regularization MTFL method [76], but bring out a significant advantage over it, in terms of selecting relevant features for classification. The new methods can simultaneously select features that are useful across multiple tasks and features that might be only discriminative for a specific classification task.

Given $T$ classification tasks in total, let $(X_t \in \mathbb{R}^{\ell_t \times d}, y_t \in \mathbb{R}^{\ell_t})$ be the sample set for the $t$-th task, where $X_t$ is a matrix containing rows of examples and columns of gait features, $y_t$ is
a column vector containing the corresponding labels for each example, \( \ell_t \) is the sample size of task \( t \), and \( d \) is the number of features. We focus on creating linear classifiers \( y_t = \text{sign}(X_t \alpha_t) \), where \( \alpha_t \) is the vector of model parameters to be determined. We then define a model parameter matrix \( A \) where each column contains a task’s parameter vector \( \alpha_t \), and thus each row of this matrix corresponds to a gait feature, i.e., the weights for a gait feature used for each of the \( T \) tasks, which we denote as \( \alpha_j \), and \( j = 1, \cdots, d \). We choose a loss function \( L(\alpha_t, X_t, y_t) \) which typically measures the discrepancy between the prediction \( X_t \alpha_t \) and the observation \( y_t \) for task \( t \). In a classification task, the loss function is commonly a logistic regression loss.

The widely used block-wise joint regularization MTFL method solves the following optimization problem for the best \( \alpha \):

\[
\min_{\alpha_t} \sum_{t=1}^{T} L(\alpha_t, X_t, y_t) + \lambda \Omega(A), \quad t = 1, \cdots, T, \tag{4.4}
\]

where \( \Omega(A) \) is a block-wise regularizer, often called the \( \ell_{1,p} \) matrix norm, that computes \( \sum_{j=1}^{d} ||\alpha^j||_p \). Common choices for \( p \) are 1, 2 or \( \infty \). Minimizing this \( \ell_{1,p} \) regularizer can shrink an entire row of \( A \) to zero, thus eliminating or selecting features for all tasks. The hyperparameter \( \lambda \) is used to play the trade-off between the loss function and the regularizer. However, a major limitation of the joint regularization MTFL method is that it either selects a feature for all tasks, or eliminates it from all tasks, which can be unnecessarily restrictive. In practice, several tasks may share features but some features may only be useful for a specific task. Hence, we introduce the following multiplicative MTFL that addresses this issue.

A family of MMTFL methods can be derived by factorizing \( \alpha_t = c \odot \beta_t \), where \( \odot \) computes a vector whose \( j \)-th component equals the product of \( c_j \) and \( \beta_t^j \), and in other words, \( a_t^j = c_j \beta_t^j \).

The vector \( c \) is applied across tasks, indicating whether certain features are useful to any of the tasks, and \( \beta_t \) is only relevant to task \( t \). We relax the indicator vector \( c \) (i.e., a binary vector) into a non-negative \( c \) so the optimization problem can be tractable. If \( c_j = 0 \), then the \( j \)-th feature will not be used by any of the models. If \( c_j > 0 \), then a specific \( \beta_t^j = 0 \) can still rule out the \( j \)-th
feature from the \( t \)-th task. We minimize a regularized loss function with separate regularizers for \( c \) and \( \beta_t \) as follows for the best models:

\[
\min_{\beta_t, c \geq 0} T \sum_{t=1}^{T} L(c, \beta_t, X_t, y_t) + \gamma_1 \sum_{t=1}^{T} \|\beta_t\|_p^p + \gamma_2 \|c\|_k^k,
\] (4.5)

where \( \|\beta_t\|_p^p = \sum_{j=1}^{d} |\beta_j|^p_t \) and \( \|c\|_k^k = \sum_{j=1}^{d} (c_j)^k \), which are the \( \ell_p \)-norm of \( \beta_t \) to the power of \( p \) and the \( \ell_k \)-norm of \( c \) to the power of \( k \) if \( p \) and \( k \) are positive integers. The tuning parameters \( \gamma_1 \) and \( \gamma_2 \) are used to balance the empirical loss and regularizers. According to the different choices of \( p \) and \( k \), we can have different levels of sparsity for \( c \) and \( \beta_t \).

The method MMTFL(2,1) refers to the case when \( p = 2 \) and \( k = 1 \) in Eq.(4.5) and solves a problem as follows:

\[
\min_{\beta_t, c \geq 0} T \sum_{t=1}^{T} L(c, \beta_t, X_t, y_t) + \gamma_1 \sum_{t=1}^{T} \|\beta_t\|_2^2 + \gamma_2 \|c\|_1,
\] (4.6)

It is widely known that \( \ell_2 \)-norm is not sparsity-inducing, meaning that minimizing it leads to a vector of many small, non-zero entries. On the other hand, the sparsity-inducing \( \ell_1 \)-norm creates a vector with many entries equal to zero. In Eq.(4.6), \( c \) is regularized by a sparsity-inducing norm, hence tending to eliminate many features from across all of the tasks. This formulation is more suitable for capturing the feature sharing pattern such that there exists a large subset of irrelevant features across tasks, requiring a sparse \( c \), but different tasks share a significant amount of features from the selected feature pool as indicated by \( c \), thus requiring a non-sparse \( \beta_t \).

The method MMTFL(1,2) is on the opposite direction when \( p = 1 \) and \( k = 2 \) in Eq.(4.5), and solves the following problem:

\[
\min_{\beta_t, c \geq 0} T \sum_{t=1}^{T} L(c, \beta_t, X_t, y_t) + \gamma_1 \sum_{t=1}^{T} \|\beta_t\|_1 + \gamma_2 \|c\|_2^2.
\] (4.7)

Eq.(4.7) is suitable to capture a feature sharing pattern where none or only a small portion of the features can be removed because each may be useful for some tasks, thus requiring a non-sparse
c. However, different tasks share a small amount of these features, thus requiring a sparse $\beta_t$. In this case, $\ell_1$-norm is applied to $\beta_t$ and $\ell_2$-norm is applied to $c$.

Since it is difficult to prove any relationship between gait features and actual gait problems, we hypothesize that these methods can help us identify the important gait features to recognize abnormal gaits due to the neurological diseases from otherwise healthy gaits, and may further locate features to discriminate between stroke-induced gaits and PD-induced gaits. To validate this hypothesis, in our performance evaluation, we compare the two methods against early MMTFL methods that are most comparable to the proposed methods and two baseline methods - single task learning (STL) methods that either use $\ell_2$-norm or $\ell_1$-norm to regularize individual $\alpha_t$, which we referred to as STL-ridge and STL-lasso respectively.

### 4.3 Performance Evaluation

We designed two sets of experiments to evaluate the effectiveness of the proposed methods. In the first set of experiments, we examined the area under the curve (AUC) classification performance metric of the models that are created by the different MTFL methods. In the second set of experiments we studied the importance of each proposed gait feature and their relevance to each classification task. In the following, we first describe our human participant study design and then present the experiment details.

#### 4.3.1 Human Subject Test Design

In order to evaluate the performance of the proposed algorithms, we collected GCF data, using the developed smart shoes, from healthy subjects without known walking problems and from PD and post-stroke patients. Experiments with healthy subjects were conducted in the Mechanical Systems Control Laboratory at the University of California, Berkeley. The clinical study with patients was conducted in the William J. Rutter Center at the University of California, San Fran-
cisco (UCSF). The Committee on Human Research (CHR) at UCSF reviewed and approved this study. The original purpose of this human subject study was to examine whether patients could use visual feedback to direct their rehabilitation training and how was the training performance compared to traditional rehabilitation training directed by a physical therapist only. We use these datasets to evaluate the algorithm developed in this chapter. Detailed experimental design and statistical analysis of the clinical outcomes are available in [17, 122].

To collect data for this work, the subjects were asked to walk multiple trials on a flat ground for at least 50 consecutive steps in their normal walking speeds. The data collected from five PD patients, three post-stroke patients, and three healthy subjects are used to test our methodology. The average ages for each of the groups are 69.2, 53 and 23 years old respectively. Representative raw data from each of the three groups are shown in Fig. 3.2. Gait features are extracted for each gait cycle and average results are taken for each trial. This generates a dataset of 180 observations with 21 features each.

### 4.3.2 Classification of Gait Disorders

To classify among stroke, PD and healthy gaits we designed and evaluated 3 classification tasks: healthy v.s. stroke gait, healthy v.s. Parkinson’s gait and stroke v.s. Parkinson’s gait. We compared our two new formulations MMTFL\{2,1\} (Eq. (4.6)) and MMTFL\{1,2\} (Eq. (4.7)) with two other standard MMTFL methods, i.e. MMTFL\{1,1\} (Eq. (4.5) with $p = k = 1$) and MMTFL\{2,2\} (Eq. (4.5) with $p = k = 2$).

In addition, two single task learning (STL) approaches were implemented as baselines and compared with the MTFL algorithms. They can be formulated as folows:

$$
\min_{\alpha_t} \sum_i ||y_t^i - X_t^i\alpha_t|| + \lambda \Omega(\alpha_t), \quad t = 1, \cdots, T, \quad (4.8)
$$

With $X_t^i$ and $y_t^i$ the i-th example and example label for task $t$ respectively, $\alpha_t$ the parameter vector for task $t$, $\lambda$ the hyperparameter used to play the trade-off between the least squares loss
and the regularizer and $\Omega$ the selected regularizer. They are summarized as STL-lasso with $||a_t||_1$ as the regularizer and STL-ridge with $||a_t||_2^2$ as the regularizer.

Before we ran the experiments we used a tuning process to find appropriate values for the hyperparameters, $\gamma_1$ and $\gamma_2$. Grid search with three-fold cross validation (CV) was performed using the training dataset to select proper hyperparameter values in the range from $10^{-3}$ to $10^3$. In all the experiments, hyperparameters were fixed to the values that yielded the best performance in the CV.

In the first set of experiments, we partitioned the 180 observations into a training dataset and a testing dataset according to a given partition ratio, which was set to be 16%, 20%, 25%, 33% or 50%, respectively in each experiment. For each partition ratio, 10-fold CV was performed and average results were reported. The classification performance was measured using AUC, which measures the total area under the receiver operating characteristic (ROC) curves. These results are summarized in the left half of table 4.2. We can observe from the results that MTFL methods always outperform STL methods. Specifically, with the smallest training set of 16%, the MMTFL$\{2,1\}$ method has the best improvement over the STL methods. When the training partition ratio was increased, the AUC performance of all the methods improved consistently. When it reached 50%, STL or MTFL methods achieved their highest AUC scores, respectively. The advantage of MTFL methods with smaller training set ratios is explained because they can learn the tasks jointly and not exclusively, which is typically done in STL methods. On the other hand, along with the increase of training dataset percentage, more training examples are provided to the classifiers, making the classification easier and thus STL methods performed closer to MTFL when the partition rate increases.

Following that, we tested how well the classification generalizes when a new subject’s gait was tested against a model built by gaits of other patients and healthy subjects. Specifically, the same classification tasks were performed with the same classification methods, but the testing data were from a single subject and all the data from the rest of subjects were used to train the corresponding model. We repeated this for each individual patient and healthy subject and
The performance results are summarized in the right half of Table 4.3, where average AUC is reported across all tasks and per task separately. PD, ST and H refer to the gait from PD patients, post-stroke patients and healthy subjects, respectively.

As can be observed from the right half of Table 4.3, MTFL methods performed better than STL methods consistently. We also observe that there were some easier tasks (e.g., stroke vs healthy), where STL AUC scores were almost as good as MTFL ones, and some more challenging tasks (e.g., PD vs healthy), where STL AUC scores were worse compared to any other task.

To further study how the two new MTFL formulations perform on each task we report the confusion matrices of all the three tasks for MMTFL\{1,2\} and MMTFL\{2,1\} in Table 4.4 and 4.5 respectively. Each row in the matrix corresponds to which gait class was tested, while a column corresponds to which gait class the algorithm predicted. Between these two new formulations, MMTFL\{1,2\} performed better with PD, as out of the 83 tested gaits, MMTFL\{1,2\} predicted 5
Table 4.4: Confusion Matrices of MMTFL{1,2} for the 3 tasks, true labels in rows, predicted in columns

<table>
<thead>
<tr>
<th></th>
<th>PD</th>
<th>Healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD</td>
<td>78</td>
<td>5</td>
</tr>
<tr>
<td>Healthy</td>
<td>5</td>
<td>59</td>
</tr>
<tr>
<td>Stroke</td>
<td>20</td>
<td>11</td>
</tr>
<tr>
<td>Healthy</td>
<td>0</td>
<td>64</td>
</tr>
<tr>
<td>Stroke</td>
<td>25</td>
<td>6</td>
</tr>
<tr>
<td>PD</td>
<td>7</td>
<td>76</td>
</tr>
</tbody>
</table>

Table 4.5: Confusion Matrices of MMTFL{2,1} for the 3 tasks, true labels in rows, predicted in columns

<table>
<thead>
<tr>
<th></th>
<th>PD</th>
<th>Healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD</td>
<td>72</td>
<td>11</td>
</tr>
<tr>
<td>Healthy</td>
<td>2</td>
<td>62</td>
</tr>
<tr>
<td>Stroke</td>
<td>28</td>
<td>3</td>
</tr>
<tr>
<td>Healthy</td>
<td>0</td>
<td>64</td>
</tr>
<tr>
<td>Stroke</td>
<td>24</td>
<td>7</td>
</tr>
<tr>
<td>PD</td>
<td>3</td>
<td>80</td>
</tr>
</tbody>
</table>

The last set of experiments aimed to report the prediction results per patient, in order to give complete information of the performance of each subject’s gait. Table 4.6 summarizes the per patient confusion matrices generated from MMTFL{2,1} for the three classification tasks. The first column indicates each subject’s disease or healthy condition and their identification numbers (ID) are given in the second column. The last two columns give number of times gait samples from the corresponding subject were predicted to be in PD, stroke or healthy gait class. The summation of these two numbers in each row corresponds to the total number of trials that were recorded for each subject. From the table we observe that stroke patient 4 was almost always predicted either healthy subject or PD patient, which means that her gait patterns were much different from the other post-stroke patients. This patient was a 33 year old female with minor stroke, which explains the similarity of her gait to healthy gait, when compared to other older stroke patients. This wrong prediction may also be related to the limited number of stroke patients that participated in this study.
<table>
<thead>
<tr>
<th>Subject</th>
<th>Predicted Disease ID</th>
<th>PD</th>
<th>Healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD 1</td>
<td>PD</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>PD 2</td>
<td>PD</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>PD 3</td>
<td>PD</td>
<td>13</td>
<td>5</td>
</tr>
<tr>
<td>PD 5</td>
<td>PD</td>
<td>19</td>
<td>0</td>
</tr>
<tr>
<td>PD 6</td>
<td>PD</td>
<td>13</td>
<td>0</td>
</tr>
<tr>
<td>Healthy 7</td>
<td>Healthy 7</td>
<td>1</td>
<td>22</td>
</tr>
<tr>
<td>Healthy 8</td>
<td>Healthy 8</td>
<td>1</td>
<td>21</td>
</tr>
<tr>
<td>Healthy 9</td>
<td>Healthy 9</td>
<td>0</td>
<td>19</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Subject</th>
<th>Predicted Disease ID</th>
<th>Stroke</th>
<th>Healthy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD 1</td>
<td>PD</td>
<td>0</td>
<td>16</td>
</tr>
<tr>
<td>PD 2</td>
<td>PD</td>
<td>0</td>
<td>17</td>
</tr>
<tr>
<td>PD 3</td>
<td>PD</td>
<td>1</td>
<td>17</td>
</tr>
<tr>
<td>PD 5</td>
<td>PD</td>
<td>2</td>
<td>17</td>
</tr>
<tr>
<td>PD 6</td>
<td>PD</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>Stroke 4</td>
<td>Stroke 4</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>Stroke 10</td>
<td>Stroke 11</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>Stroke 11</td>
<td>Stroke 11</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

**Table 4.6: Confusion Matrices of MMTFL\{$2,1$} for the 3 tasks per patient**

Given that MMTFL\{$2,1$} performs best in general, the tested data seem to follow the assumption under which MMTFL\{$2,1$} was designed. Specifically, across all three tasks there exists a large subset of irrelevant sensing features, requiring a sparc\$c\$, but different tasks share a significant amount of features from the selected feature pool as indicated by \$c\$. In other words, there are some specific sensing features that help identify the neurological gait disorders. Next, we present the important features selected by each method.

### 4.3.3 Identification of Important Gait Features

Important gait features identified from gait disorder classification may help better understand the key characteristics that distinguish abnormal gait patterns among different gait disorders and healthy gait. They may also help the target design of treatment and evaluation of rehabilitative progress. In this subsection we present the important gait features that were identified by the used methods in our experiments, for each of the three classification tasks that were evaluated in.
With the important gait features we can understand which of the proposed gait features are more important to classify GCF data from post-stroke or PD patients and healthy subjects. As described in section 4.2 for the MMTFL methods, we have $\alpha_t = c \odot \beta_t$. Vector $\alpha_t$ is the vector of model parameters for task $t$, $c$ vector is used across all tasks, indicating if a feature is useful for any of the tasks, and vector $\beta_t$ is only for task $t$. In Fig. 4.2 we plot all vectors $c$ for each MMTFL model as progress bars to show the importance of each feature. In Fig. 4.3 we plot the absolute value of the task parameter vector $\alpha_t$ for each MMTFL and STL method for the stroke against healthy classification task. Additional task parameter vectors for the other two tasks can be seen in a technical report of this work in [81]. Based on the general characteristics of Hemiplegic gait, most commonly seen in stroke, and Parkinsonian gait [28] we have the following observations.

First, the two most important features are maximum force at the right toe and maximum force at the left heel. These two are strength indicators during toe off and heel strike gait phases. Patients with neurological related diseases, like stroke and PD, may experience weak muscle strength [28]. Circumduction of the affected leg in stroke can also produce different toe contact force signatures. Additionally, slow walking (Bradykinesia) which is characteristic of both stroke and PD gait can have reduced force levels at the toe during push-off [28, 118].

Minimum force difference between medial and lateral sides of the metatarsophalangeal joints at the forefoot (see Sec. 4.1.5) at the left foot is another important feature, which is an indicator of balance. Rigidity, meaning stiff or inflexible muscles, is one of the main symptoms of PD, alongside tremor and slowness of movement. There is usually little or no arm swing to help in balancing the individual [28]. PD patients usually have reduced balance and the algorithm has identified this as an important feature.

Cadence and double support ratio are mobility gait features and they are also important in distinguishing healthy vs pathological gait. As discussed before, a common characteristic of stroke and PD subjects’ gait is bradykinesia. This in turn affects the double support ratio.

Finally, symmetry of swing phases is found to be another important factor to distinguish
Figure 4.2: Feature selection vector \( \mathbf{c} \) from all MMTFL methods
Figure 4.3: Absolute value of task parameter vector $\alpha_t$ in the Stroke vs. healthy gait classification task.
pathological gaits for some models. This parameter captures how evenly the swing gait phases are represented in the subject’s gait. Circumduction of the affected leg in hemiplegic gait can introduce additional gait phases, which can affect the gait symmetry.

All the rest features are not important and discarded by most of the models, except MMTFL\{2,2\}, which shows reduced sparsity. These findings are consistent with the literature about the characteristics of PD and stroke patient’s gait [28].

4.4 Conclusion

In this chapter, we presented the design of an integrative framework for gait disorder diagnosis and advance smart gait rehabilitation. Gait features were developed for different categories including gait phases, mobility, balance and strength. MTFL, an advanced classification method, was used to train the different classification tasks that can classify subject’s gait. Data from PD and post-stroke patients, along with healthy subjects were used to evaluate the proposed methods. The proposed gait features successfully captured the underlying properties of each disease. MTFL was able to construct accurate classifiers based on the given gait parameters to distinguish abnormal gaits. Also, it selected the most important gait parameters for each classification task, ignoring the rest. Selected features captured the characteristics of each disease as described in the literature. This study demonstrated the potential to automate gait analysis of multiple gait disorders, which can benefit the medical professionals and patients with improved and targeted treatment plans for rehabilitation.

Apart from gait deviations that are caused from neurological abnormalities, it has been shown that unique individual characteristics can cause differences in gait patterns observed in different humans. In order to further extend this framework to identify such individual gait deviations and differences, the next topic investigated in this thesis is the development of a new method that improves the state of art in biometric authentication based on gait patterns. By taking advantage of the individual variations in gait patterns, the presented framework is designed to combine...
multiple sensing modalities that can achieve performance improvement compared to the state of art.
Chapter 5

Gait based authentication

Passwords and keys allow users to access their personal information, while protecting against unauthorized attempts. However, studies have shown that users often choose weak and easy to remember passwords like “12345”, “abc1234” or even “password” to protect their data, even though those passwords are easy for an unauthorized user to guess [85]. Strong passwords that combine characters, numbers and symbols are more difficult to hack but can be easily forgotten.

In order to bridge the gap between secure authentication and usability, there has been a shift towards biometric authentication methods which take advantage of biological features, such as fingerprints and face characteristics [27], or behavioral features like speech, keystroke dynamics [13], swipe patterns [85] and gait patterns [29, 47, 50, 72, 100]. Those features cannot be forgotten and thus biometric authentication methods significantly improve the usability. Furthermore, continuous biometric authentication systems, especially on mobile devices, are gaining more interests in recent years. Instead of authenticating the user only at the entry point when the device is locked, those systems determine whether biometric traits correspond to a respective user in a real-time and continuous manner. In this way, users can be continuously monitored after initial access and thus do not need to constantly worry about security and privacy in case their devices are lost [85].

Gait-based authentication is among the most popular behavioral biometric authentication
methods. Gait refers to locomotion achieved through the movement of limbs and due to the different properties of an individual’s muscular-skeletal structure, gait patterns are fairly unique among individuals [125]. Gait-based continuous authentication seeks to verify whether the user is genuine in a periodic or constant manner without interrupting the user’s normal interaction. It requires little cooperation from the user, and is usually an inexpensive option. Gait is also difficult to mimic [36, 65, 66], making spoofing of gait a hard task for an adversary. To perform gait-based authentication, multiple technologies have been used such as cameras for computer vision-based gait recognition [21, 42], or smart mats [82] and plates [30] for floor sensing. With the rapid development of wearable sensors and mobile phones, an increased amount of works that utilize those technologies has been performed recently. However, the majority of those devices are equipped with an inertial measurement unit (IMU) and a common issue of IMU measurements is the lack of accuracy, robustness and reliability [100].

Motivated by these recent technological advances, in this chapter, we present a gait-based continuous authentication framework using multimodal learning. Specifically, our approach aims to support a more user friendly and robust authentication method by combining two sensing modalities, i.e., accelerometer (ACC) data and ground contact force (GCF) data. We employ a multimodal learning approach based on autoencoders to explore the relationships between these two different modalities of the data and thus build more robust learning models leading to more accurate authentication results. Two types of sensor fusion techniques are explored, i.e. early and late sensor fusion. Early fusion is based on the hypothesis that it is possible to develop models that use simple time-domain features for authentication, while the hypothesis for the late sensor fusion is that more complex and abstract features are required for gait-based authentication, and thus extraction of higher-order features based on simple time-domain features is required.

The proposed authentication method can be used in broad application scenarios. Specifically, it may be used as part of a multi-factor authentication framework on mobile phones [67], which is considered stronger than single factor or multi-layer authentication [8]. It may also be used in situations where other strategies such as facial recognition and fingerprints cannot be applied.
Furthermore, it can help enhance the UI experience, through adaptive interfaces based on different active users, or it can be used along with access control schemes, where access to more secure features of the device can be done with other more secure biometric methods or strong passwords. In this way, usability is increased, while security is kept high.

The effectiveness of our approach is evaluated through extensive experiments on datasets collected from two case studies, one with commercial off-the-shelf (COTS) running smart socks and the other with a research prototype of smart shoes designed for lower-extremity rehabilitative training, both of which can record GCF and ACC data. With the use of these two different sensing platforms we can evaluate the generalizability of our approach. In the experiments, we first evaluate the robustness of the proposed authentication framework under different attack scenarios, such as passive attack and active attack, i.e. when impostors perform gait mimicking while observing their victim’s gait in real time. On top of that, we further evaluate the robustness of the proposed approach while controlling two parameters that effect gait patterns, i.e. walking speed [53] and fatigue [43, 89]. Finally, a per modality and per fusion technique evaluation is performed, based on individual sensing platforms. Our extensive results show that between the two modalities used in this approach, GCF is more robust than ACC. In addition, our evaluation shows that an early fusion of the ACC and GCF modalities is the most robust approach, compared to the GCF modality only or any other fusion method. By utilizing information in ACC and GCF data, the models can achieve equal error rates (EER) of as low as 0.01% for the smart socks platform and 0.16% for the smart shoes platform. The leave-one-out approach, which evaluates the generalizability of the proposed method when a never-seen-before impostor tries to be authenticated, achieves a false acceptance rate (FAR) of 0.54% for smart socks and 1.96% for smart shoes. Walking parameters, such as speed and fatigue from everyday activities are shown to have significant impact on the authentication performance as well. This suggests that providing diverse gait samples can further improve the robustness of gait-based authentication models.

The remainder of this chapter is organized as follows. Section [5.1] gives an overview of
Figure 5.1: An overview of the proposed integrative framework for multimodal gait-based continuous authentication: (a) sensing platforms, (b) filtering/cycle extraction, (c) feature extraction, (d) classification, and (e) authentication.

our approach. The two sensing devices used for data acquisition are discussed in Section 5.2. Section 5.3 describes our filtering and data segmentation techniques. Section 5.4 discusses the feature extraction and classification models of the gait patterns. Section 5.5 describes the data collection protocol and the evaluation results. Finally, we conclude the chapter in Section 5.6.

5.1 Methodology overview

To support multimodal and continuous gait-based authentication we propose an integrative framework that comprises of a data acquisition platform, a gait cycle extraction component, a feature extraction component based on autoencoders and a classification component. An overview of the proposed framework is presented in Fig. 5.1. The sensing device used in the data acquisition platform (Fig. 5.1a) is either a pair of smart socks or smart shoes (see Section 5.2 for the details). Both sensing devices are capable of recording synchronized GCF and ACC motion data, and are equipped with wireless modules to transfer the recorded data to an application on a phone or a laptop. The corresponding wireless connection between the sensing platform and the mobile phone is considered to be secure and encrypted, so that there is no possibility for a replay attack, i.e. a type of attack in a biometric system where old captured data from previous sessions or other users are used for authentication.
The collected data samples are then sent to the filtering and gait cycle extraction component (Fig. 5.1b) which is responsible for filtering the ACC data, segmenting the GCF and filtered ACC gait data into individual gait cycles, and storing the data within each gait cycle in a vector and forward them to the feature extraction component (Fig. 5.1c). For feature extraction, we use autoencoders. Different sensor fusion techniques can be achieved, depending on the way how the selected autoencoders are connected and what sensor types they get their input from. An overview of the five investigated sensor-fusion techniques is summarized in Fig. 5.5. The first model receives the stacked raw ACC and GCF gait cycle data as input and learns a shared representation. The next two models each use only raw gait cycle data from one of the two modalities. The fourth model stacks the two individual encoder outputs from each modality. Finally, the last model forms a bimodal-deep autoencoder network that learns higher order features [71]. The number of nodes in each autoencoder has been empirically selected, aiming to reduce the initial raw data dimensionality, and improve the feature quality and class separation (see Section 5.3 for the details).

The extracted features are finally sent to the last component of the processing pipeline for classification (Fig. 5.1d). The classifier decides whether the given feature vector from the corresponding gait cycle belongs to the legitimate user or not. In order to achieve this, we build classification tasks for each individual user. Training data belonging to the corresponding owner of the model are marked to be in the positive class, while data from all the other subjects in the training set are marked to be in the negative class. In order to achieve continuous gait-based authentication, the pipeline can be repeatedly invoked to decide whether a given gait cycle belongs to the corresponding owner of the mobile device. If the device is unlocked there could be a time interval before the method is invoked again to save resources, while keeping the user authenticated continuously. In addition, to improve the overall user experience and authentication performance, an additional ensemble or voting layer can be used to decide whether to lock or unlock the device based on the classification results from a number of past gait cycles. In this chapter, we focus on the first four components of the proposed framework as depicted in Fig.
and plan to extend our study on the additional layers in the future work. The following three sections give a more detailed description of the components in the proposed gait authentication framework.

## 5.2 Data Acquisition Platforms

Multimodal learning essentially reveals correlations among different modalities from multiple data sources to build stronger and more robust learning models than those learned from individual modality [71]. In this work, we relate features extracted from users’ ACC and GCF data to build an improved model for gait-based behavioral biometric authentication. To demonstrate that our algorithms can be applied on different sensing platforms, we use both a commercial off-the-shelf (COTS) smart socks and a research prototype smart shoes for data acquisition. As will be shown in Section 5.5, despite the numerous differences between the two sensing devices, our algorithms can achieve similar performance and are not affected by the different characteristics of the sensing platforms.

The COTS smart socks are purchased from Sensoria [94] (see Fig. 5.2a). The socks are designed for runners who need to improve their running skills and get real-time feedback on multiple parameters, such as cadence, foot landing position, pace and speed. The Sensoria Software Development Kit (SDK) includes a license to support raw data collection, a pair of smart socks and a pair of Bluetooth anklets for wireless data collection. The socks are embedded with 3 proprietary textile pressure sensors, attached at the bottom of the sock, one in the heel area under the calcareous bone and two in the metatarsal area, at the first and fifth joints, respectively. The collected GCF signals are relayed through conductive fibers to the anklet. The attachable Bluetooth anklet contains a 3-axis accelerometer, making the hardware completely mobile. The GCF signals along with the ACC signals, are sent to the SensoriaLab iOS application, where data can be stored locally or uploaded to the cloud for further processing. The battery of the anklet allows about 6 hours of operation and the socks’ sampling frequency is set to 32Hz.
Figure 5.2: The two sensing platforms for data acquisition.

The smart shoe is a novel wireless human motion monitoring system for gait analysis in rehabilitation training [122]. It is developed to measure the GCF at four points: toe, first metatarsal joint (Meta1), fourth metatarsal joint (Meta4) and heel, as shown in Fig. 5.2b. The silicone tubes are wound into air bladders and connected to barometric pressure sensors. In addition, an IMU sensor is attached to the distal end of the shank to measure the accelerations and rotations in three dimensions. The sampling rate of the system is set to 30 Hz and the data is sent to a high-performance laptop through WiFi. It should be noted that the GCF measurements from both platforms are affected by the material properties of the shoe bottom and the conditions of the floor. Since all of the participants are required to wear the shoes with soft and compliant bottom, we hope to minimize the affect of the shoe bottom in gait patterns. Also, we assume that the participants will walk on a non-slippery and stiff floor such that this floor factor can be ignored.

Despite the fact that both sensing devices provide similar sensing modalities, there are multiple differences between the two. For example, the smart socks are equipped with three textile pressure sensors at the bottom of the socks, while the smart shoes have four air bladders embedded in the sole and connected to the barometric sensor on the back. In the smart socks, the ACC data are collected by the attachable Bluetooth anklet, while in the smart shoes, the IMU sensor measurement is sent through the WiFi. These characteristics can cause evident differences in the collected raw data. For example, in Fig. 5.3b and 3d, we can observe that the data collected from smart shoes change more sharply than the measurements from the socks. This could be due to
Figure 5.3: Raw ACC and GCF data from both the smart shoes and Sensoria socks.
the participants’ walking patterns since the data are collected from two individuals. In addition, a second factor for this difference could be the nature of the textile material which is less sensitive, compared to the barometric sensors that can capture small variations in data.

5.3 Data Filtering and Gait Cycle Detection

After receiving the ACC and GCF measurements from the data acquisition platforms, a pipeline of processing components are employed in order to provide continuous and robust gait-based authentication. In this section, we examine the data filtering and gait cycle detection components, which are used to reduce the noise levels in the ACC signals and segment the data stream so that further processing can be performed.

5.3.1 Filtering

Both sensing devices provide raw unfiltered ACC and GCF data. As it can be seen in Fig. 5.3a and Fig. 5.3c the ACC data contain higher noise levels compared to the GCF data. In order to reduce the noise levels in the ACC data, we employ a moving average approach. Moving average filters are low-pass filters, which are easy to implement, and provide great smoothing performance. This greatly helps in the following analytic layers to prevent overfitting and improve generalization. In our design, the ACC raw data from every sensor channel (x, y and z directions) from both sensing devices will pass through a 5-point moving average filter. The filter length was chosen to be small as we do not want to miss any important information, such as spikes from the feet movement, that could help distinguish a subject. The filter output is given by the following difference equation:

\[ y[i] = \frac{1}{N} \sum_{j=0}^{N-1} x[i-j] \]  

(5.1)

where \( y[i] \) is the filter output at timepoint \( i \), \( x[i] \) represents the input data at timepoint \( i \), and \( N = 5 \) is the filter length. If \( i < j \) we can set \( y[i] = x[i] \) and skip filtering the first \( N \) observations.
5.3.2 Gait Cycle Detection

Detecting gait cycles (Fig. 2.1) can be challenging, especially when only ACC data are used and the IMU sensor recording point is far away from the foot. However, the heel GCF sensor data can help easily detect heel strikes. By using the measured contact force of the heel with the ground, we can accurately detect the repeating heel strike gait phases. Based on this, we can define a gait cycle to be the time interval between two consecutive left heel strikes. Since the gait cycle length is not constant and depends on the walking speed, we use a fixed-size window that starts with a left foot heel strike, and has a length of 37 samples, i.e. 1.156 seconds at 32Hz or 1.233 seconds at 30Hz (Fig. 5.1b). This window length is empirically chosen to capture any walking speeds that a subject may be walking with, except extremely slow walking, such as walking with less than 48 strides per minute. Note that on the two data acquisition platforms, both ACC and GCF data are recorded with the same timestamp. Thus data synchronization between the two modalities is not necessary during the gait cycle extraction.

Once gait cycles have been identified, normalization across the two different modalities is performed, so that normalized data lie in the \([0, 1]\) space. Normalization is important not only to help utilize the autoencoders in all the models for feature extraction, but also to eliminate differences between body-weight across individuals. In this way adversaries may not benefit when they try to match their body-weight to their victim. The formula used for normalization for both ACC and GCF data is as follows.

\[
y_j[i] = \frac{x_j[i] - \min x_j[i]}{\max x_j[i] - \min x_j[i]}, \quad i \subseteq I, \quad j \in C
\]  

(5.2)

\(I\) refers to the set of indices, \(i\), that belong to one gait cycle, while \(C\) refers to all the channels that correspond to this modality. By taking the \(\min\) and \(\max\) across all the channels in a gait cycle, we are able to keep the relative differences across sensor channels, and thus let the models used at the later components benefit from differences in force levels, or timing of gait patterns.
The last step of gait cycle detection is to concatenate the feature vector into a single feature vector. The vector length is $L_v = L_w \times 2 \times N_c$, where $L_w = 37$ (observations/window) is the window length, 2 is the number of feet (left and right), and $N_c$ is the number of channels for each sensor. For example, the ACC sensors have 3 channels (x, y and z directions), the shoe GCF sensors have 4 channels (heel, meta12, meta45 and toe) and the sock GCF sensors have 3 channels (heel, meta12 and meta). This results in a vector of 222 input features for the ACC modality for both smart socks and shoes, 222 features for the GCF modality of the smart socks and 296 features for the GCF modality of the smart shoes. Before these gait cycle vectors are sent to the next component for feature extraction, in order to train clean models and improve generalizability, outliers removal is performed. If the corresponding gait cycle vector contains more than 10 features that have observations more than 3 standard deviations away from their mean across all the gait cycle vectors belonging to the corresponding user, this feature vector is considered outlier and will be discarded.

### 5.4 Feature Extraction and Classification of Gait Patterns

To extract features from the detected gait cycles we employ autoencoders. Autoencoders are used as a building block for early and late fusion of ACC and GCF data within a gait cycle. Early fusion technique tries to learn models that extract simple temporal and amplitude features, while late fusion technique uses per modality features to extract higher order and abstract features. In the following, we first describe the detailed characteristics of autoencoders and then discuss how they are used for feature extraction and sensor fusion. Finally, we describe the classification algorithms used on the extracted features to support gait-based authentication.
5.4.1 Feature Extraction with Auto-encoders

Autoencoders have recently been applied in a broad range of applications (e.g., image, video and audio processing [71]) to find higher order correlations from different data sources and extract meaningful features that can better represent the data. Compared to linear methods such as the principal component analysis (PCA), autoencoders can achieve better feature extraction and dimensionality reduction, due to its non-linear transfer function. For these reasons, in this work we use autoencoders to extract features separately for each vector from the two different modalities and then add another layer for bimodal feature extraction.

An auto-encoder is a neural network that is trained to replicate its input at its output (see Fig. 5.4 for a conceptual structure). Training an autoencoder is unsupervised, in the sense that no labeled data is needed, and is based on the minimization of the error between input $x$ and its reconstruction at the output $\hat{x}$. An autoencoder is composed of an encoder and a decoder. Given an input vector $x \in \mathbb{R}^D$, the encoder tries to map vector $x$ to a hidden representation $z \in \mathbb{R}^{D'}$ by learning a function $h_{W,b}$ as follows: $z = h_{W,b}(x) = s(Wx + b)$. $s$ is a transfer function for the encoder (we use the sigmoid function), $W \in \mathbb{R}^{D' \times D}$ is a weight matrix, $b \in \mathbb{R}^{D'}$ is a bias vector, and $D, D'$ are the number of nodes at the input and hidden layers, respectively. During training the weight matrix $W$ and the bias vector $b$ are learned.

The decoder maps the encoded representation $z$ back to a reconstructed vector $\hat{x}$ in input space by learning a function $g_{W',b'}$ as follows: $\hat{x} = g_{W',b'}(z) = s'(W'z + b')$, where $W' \in \mathbb{R}^{D \times D'}$ is a weight matrix, and $b' \in \mathbb{R}^D$ is a bias vector. Autoencoders can achieve better feature extraction and dimensionality reduction, compared to linear methods (e.g., PCA) due to the non-
linear transfer function $s$.

The loss function used to train autoencoders is typically the mean squared error loss:

$$E = \frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{D} (x^j_n - \hat{x}^j_n)^2,$$

(5.3)

where $N$ is the number of observations, and $D$ is the number of variables in the training data, $x^j_n$ is the $j$-th variable of the $n$-th training sample, and $\hat{x}^j_n$ is the $j$-th variable of the reconstruction of the $n$-th training sample from the autoencoder. To avoid over-fitting, a regularization term $\Omega_w$ (weight decay) is typically introduced which favors small weights in $W$. In addition sparsity on the encoded representation can be enforced by adding a regularization term $\Omega_s$ that takes a large value when the average activation value, $\hat{\rho}_i$, of a neuron $i$ and its desired value, $\rho_i$, are not close in value [77]. By using the autoencoders as a building block, we can develop different models that perform feature extraction with different characteristics. We discuss the employed approaches in the following subsections.

### 5.4.2 Early and Late Sensor Fusion Techniques

When dealing with multimodal sensor data in neural networks, it is possible to fuse the data at different stages of the network, achieving different sensor fusion techniques [70]. Here we investigate early fusion and late fusion approaches, while also evaluating no-fusion between modalities to understand the importance of GCF and ACC modalities. Early fusion has the advantage of a simple model, that lets the algorithm figure out which feature from which modality is important for authentication. On the other hand, late fusion is designed to identify more abstract characteristics, as it is the task of the lower layer encoders to learn specific features from each modality.

An overview of the investigated sensor fusion techniques is presented in Fig. 5.5. The first model (Fig. 5.5a) performs early fusion by receiving as input a stacked vector of raw ACC
and GCF gait cycle data and treating them equally. Its advantage lies in extracting temporal and amplitude related features that may be easier to detect with a simple model. The next two models (Fig. 5.5b and 5.5c) each uses only raw gait cycle data from one of the two modalities, performing no sensor fusion with the other modality. They have similar characteristics with the first model in extracting simple features for each modality, but do not perform any fusion of features between the two modalities.

Late sensor fusion is performed by the last two models. The fourth model (Fig. 5.5d) is an extension of the two per-modality feature extraction models. It combines the outputs of these two models and lets the classification algorithm to learn a good combination of features extracted from the two modalities separately. Finally, the last model (Fig. 5.5e) forms late fusion with a bimodal-deep autoencoder network that fuses the two individual models at the second layer in order to learn higher order, more complex and abstract features given features that were extracted at the first layer, per modality [71]. Its advantage lies in extracting features that are not easy to detect with a single layer of encoding and may require more complex, non-linear transformations, such as those required for audio-visual classification [71].

For all the models, the decoders of any autoencoder are discarded after training as the goal is for feature extraction and no reconstruction is required. The number of nodes for each encoder is set to 25, with the exception of the raw-stacked model where 50 nodes are employed. This selection was decided after trying different number of nodes and selecting those leading to increased performance. Higher number of nodes for the raw-stacked model could be explained since this model has to fuse features from both modalities, so there could be more information potential to be learned from the raw data.

5.4.3 Classification of Gait Features

Most related work in the literature studies the authentication or identification of gait using distance metrics or pattern similarity measures, such as dynamic time warping (DTW) [101]. One
drawback of those methods is that DTW might warp the series too much so that the series lose its discriminative patterns [124]. In this work however, we use autoencoders to perform feature extraction and rely on a classification algorithm to make the final decision of authentication. It is a standard practice for many neural network algorithms to attach an additional final layer (called soft-max layer) of one output node per data class, that will provide a probability that the corresponding input belongs to each class. Here we define a data class to be a human subject, so after a new gait cycle is passed through the model, we have class probabilities, i.e. the probability that it belongs to any class, with the sum of probabilities equal to one.

In addition, we use support vector machine (SVM) algorithm, which is one of the most popular machine learning algorithms. SVM can achieve improved performance in binary classification, and especially in authentication, since the algorithm is trained to learn a separating hyper-plane that separates the positive class (genuine user) from the negative class (impostor users) in the best possible way. By doing so, we expect that unknown gait samples that belong to an unknown impostor user can be more easily rejected by SVM, which is trained to specifically identify gait patterns of the corresponding genuine user. SVM maps the input points in a high dimensional space using a kernel. A hyper-plane is used to divide the geometric space into two parts for classification. The main advantage of SVM is that it solves a convex problem and is suitable for classification of continuous features. The regularization parameters can be tweaked...
to control over-fitting. We use SVM by defining a classification task for authentication of each of the subjects, which predicts whether a new gait sample originated from that user or not. To train SVM the user’s gait is marked as positive, while all other subjects’ gait samples are marked as negative.

5.5 Performance Evaluation

To evaluate the effectiveness of the proposed gait-based authentication framework, we design and perform two case studies. We use smart shoes in one of the studies and smart socks in the other. This helps evaluate the robustness of the approach on different sensing platforms. In addition, different testing scenarios are considered for each of the studies that can affect gait dynamics, such as walking speeds and fatigue levels. In the following, we first summarize the design principles of the case studies and the experiment setup. We then present the details of the experimental results.

5.5.1 Case Studies

In both case studies, we select healthy and young adult subjects to participate in the experiments. The studies are designed to test the efficacy of the proposed algorithmic framework under different scenarios (see Table 5.1). Our first case study uses Sensoria smart socks (Fig. 5.2a). 7 female and 8 male healthy students participate in this pilot study and their ages range from 20 to 29. For each subject we demonstrate how to use the hardware and give instructions for the data collection sessions. Specifically, the subjects are asked to walk normally on a 55 feet long hallway back and forth for 5 minutes. To collect the two modalities of the gait data, the subjects wear the smart socks and carry a smart phone in their pocket with the SensoriaLab iPhone application for data storage. Before the 5 minute recording session, we make sure that the socks are worn correctly, the Bluetooth anklet is adequately charged, the wireless connection with the phone is
Table 5.1: A summary of the two case studies with different sensing platforms and testing scenarios.

<table>
<thead>
<tr>
<th>Case Study</th>
<th>Sensing Platform</th>
<th>Testing Scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>First</td>
<td>Smart socks</td>
<td>Fast walking</td>
</tr>
<tr>
<td>First</td>
<td>Smart socks</td>
<td>Slow walking</td>
</tr>
<tr>
<td>Second</td>
<td>Smart shoes</td>
<td>Fast walking in the morning</td>
</tr>
<tr>
<td>Second</td>
<td>Smart shoes</td>
<td>Fast walking in the afternoon</td>
</tr>
<tr>
<td>Second</td>
<td>Smart shoes</td>
<td>Slow walking in the morning</td>
</tr>
<tr>
<td>Second</td>
<td>Smart shoes</td>
<td>Slow walking in the afternoon</td>
</tr>
</tbody>
</table>

in good condition, and the ACC and GCF signal quality from the socks and anklet do not show any abnormalities or excessive noise. During the 5 minute recording session, the subjects are asked to walk in two different walking speeds, i.e. slow and fast walking. At the first two and half minutes they walk at slow speed, between 3-4 feet per second and at the remaining time they walk between 5-6 feet per second. In order to make sure that the subjects follow the walking speed requirements, the student researcher kept walking on the side of the subject for the first two round-trip walks for both walking speed recording sessions in order to keep the speed constant. The researcher had performed the walking session multiple times using a timer to measure the exact time requirements for the given hallway length. The sampling rate of the smart socks is 32Hz and the total number of gait cycles across all subjects for all sessions recorded is 4004.
In the second study, 10 healthy male subjects aged from 21 to 27 are invited. Limited by the shoe size, only male participants are selected, whose shoe sizes are either 10 or 11. The subjects are asked to walk on a 50 feet long hallway at two speed cases: slow and fast. The speed for slow and fast walking are set as 3-4 feet per second and 5-6 per second correspondingly. To ensure the recorded data are within the speed range, makers are labeled on the hallway floor for every 5 feet and a camera is used to record the whole experiment, shown in Fig. 5.6. After synchronizing the video with the shoe measurements, the data where the speed is not within the requirement are removed. For each subject, the experiments are finished twice at around 11 a.m. in the morning and 4 p.m. in the afternoon on the same day. The goal of this study is to investigate whether the reduced energy level in the afternoon would affect the performance of gait-based biometric authentication. In total, 9357 gait cycles are recorded from all the subjects. The data are collected by the smart shoes at a sampling rate of 30 Hz to match the experiment with smart socks. Data for joint accelerations in three dimensions are collected and pressure data at four different locations (toe, the first and second metatarsophalangeal joint, the fourth and fifth metatarsophalangeal joint, and the heel) at feet are collected.

5.5.2 Experimental Setup

To test the effectiveness of the proposed authentication method, we train a binary classifier for each of the subjects in the dataset. Data originating from the corresponding subject are marked to be in the positive class, while all the others are marked negative. To report the generalizability of the models, $k$-fold cross-validation (CV) is adopted, with $k = 5$. In $k$-fold cross-validation, the dataset is split into $k$ separate equal subsets, one subset is used for testing and the rest $k - 1$ subsets are used for training, and this is repeated for all the $k$ subsets. In addition, we perform leave-one-out cross-validation to report how well the models can generalize and predict a never-seen-before impostor. In this case, data belonging to the impostor subject are left out for testing, and the rest data from all the other subjects are used to train the models. For all the experiments,
average and standard deviation of the performance metrics are reported. The available training set for each experiment is used to train both the auto-encoders network and the classifiers. The test set is tested against the models returned by the training set.

To select the hyper-parameters for both the auto-encoders and classifiers, grid search is conducted on the complete dataset from the first case study and the set of parameters that achieved the highest performance is selected. Due to the hardware differences between the two sensing devices, experiments are done independently and only data from one sensing device are used in each experiment.

Biometric authentication methods are typically evaluated using three performance metrics, i.e. false acceptance rate (FAR), false rejection rate (FRR) and equal error rate (EER). Given a new-to-be-tested observation, a classifier returns a score (or probability) that this observation belongs to the positive class, i.e., the class with gait samples of the genuine user. If this score exceeds the acceptance threshold, the observation is accepted, otherwise is rejected. Based on that, FAR is defined as the portion of imposting recognition attempts that are accepted (score above threshold) and FRR is defined as the portion of genuine recognition attempts that are rejected (score below and equal to threshold). A trade-off between these two types of errors is achieved by varying the acceptance threshold, so that as error of one type decreases, error of the other type increases. Thus a common way of evaluating the performance of a biometric system is to estimate the point where FAR and FRR are approximately equal [112], which is called EER. In our experiments, we report EER when we perform k-fold CV, since the test set contains observations from both negative and positive classes, so we can report the point where FAR and FRR are equal. If the generated data do not provide scores that set FAR and FRR equal, we report the average of the two metrics at the point where their difference is minimum. Finally, we report FAR for the leave-one-out CV, since the test set of these experiments contains impostor gait samples and thus should be rejected by the algorithms.

We perform extensive experiments to evaluate the performance of the proposed authentication method. First we perform cross-validation and leave-one-out evaluation to assess general-
Table 5.2: EER and FAR performance per modality and fusion model, with SVM classification.

<table>
<thead>
<tr>
<th>Platforms</th>
<th>Metric</th>
<th>Raw stck</th>
<th>ACC</th>
<th>GCF</th>
<th>Stck</th>
<th>Bimodal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shoes</td>
<td>EER</td>
<td>0.16 ± 0.22</td>
<td>1.58 ± 0.96</td>
<td>0.59 ± 0.59</td>
<td>0.18 ± 0.23</td>
<td>0.37 ± 0.43</td>
</tr>
<tr>
<td>Socks</td>
<td>EER</td>
<td>0.01 ± 0.04</td>
<td>5.58 ± 4.13</td>
<td>0.01 ± 0.05</td>
<td>0.00 ± 0.00</td>
<td>0.01 ± 0.02</td>
</tr>
<tr>
<td>Shoes</td>
<td>FAR</td>
<td>1.96 ± 3.69</td>
<td>6.40 ± 6.46</td>
<td>4.48 ± 6.50</td>
<td>2.09 ± 4.04</td>
<td>2.96 ± 4.90</td>
</tr>
<tr>
<td>Socks</td>
<td>FAR</td>
<td>0.54 ± 0.23</td>
<td>10.91 ± 11.53</td>
<td>0.77 ± 3.16</td>
<td>0.56 ± 2.60</td>
<td>0.84 ± 3.56</td>
</tr>
</tbody>
</table>

5.5.3 Expt. 1: Cross-validation and Leave-one-out evaluation

In the first set of experiments, we first perform 10-fold cross-validation on both case studies. Average EER results across all folds and all subjects are reported in Fig. 5.7a. These results (SVM only) can also be seen in Table 5.2, since the small differences cannot be visualized. For this experiment, data from both slow and fast walking speeds are used for smart socks and smart shoes, and both morning and afternoon sessions are used for the smart shoes.

To evaluate the generalizability of the models, we further evaluate the effectiveness of the proposed approach when a never-seen-before impostor tries to get authenticated. For this, we perform leave-one-out cross-validation using training data similar to the previous experiment, including both slow and fast walking datasets from the socks and shoes studies and both morning and afternoon data from the shoes study. This experiment can be taken as a passive attack, since the impostors select their victim to attack without performing any active attempt to hack their gait-based authentication model. They passively hope that their gait patterns are close enough to the victim’s patterns, so that the models accept them. Both the average and standard deviation of the FAR values are reported in Fig. 5.7b. A summary of the performance per model is given in
From these results, we first observe that the GCF modality, for both studies, outperforms the ACC modality. This indicates that by adopting advanced sensor technologies that can record GCF data, we are able to significantly improve the authentication performance compared to those methods using ACC data only. In addition, by employing multimodal learning, and fusing the two modalities, we are able to further improve EER and FAR comparing to any single modality. All the fusion models outperform GCF and ACC modalities, except in the socks leave-one-out experiment (last row in Table 5.2), where GCF performs better than the bimodal fusion model.

Another observation is that authentication in the socks study seems easier (with lower EER and FAR) compared to the shoes study. This could be attributed to the difference in the subjects participated in the study and the increased resolution of the shoe design that may make the authentication task harder, as more variance to the gait patterns is introduced. In addition, differences between shoes worn are eliminated in the shoes study, since all the participated subjects wear the same pair of smart shoes to collect data. While in the socks experiment, subjects are allowed to participate with their own shoes. A detailed discussion on this follows in subsection 5.5.7.

From the results, we also observe that the raw-stacked model, i.e. the one that performs early fusion in the two modalities (Tab. 5.2), performs better compared to the other fusion models, as any classifier seems to achieve lower EER with this model. This might indicate that early fusion may be sufficient to learn simple features in the time domain and develop authentication models. In addition, the proposed approach can be successfully applied to gait identification or recognition applications, where the goal is to determine the identity of the subject, given a new observation, based on a database of gait samples from a set of enrolled known subjects.

Finally, we observe increased FAR when compared to EER of the 10-fold CV from the first experiment, in both the shoes and socks studies. This performance degradation is expected, since the test data come from a subject that the auto-encoder and classification models have never seen before. It is important to note that most of the related work on gait-based authentication fail to
Figure 5.7: 10-fold and leave-one-out CV for both socks and shoes studies

report the leave-one-out FAR. We believe this experiment should be reported to have a complete and fair evaluation on the effectiveness of the proposed approach as typically in real-life scenario an impostor will not provide his training set to the gait-based authentication system.

A per-subject leave-one-out FAR performance evaluation is given in Fig. 5.8 for both case studies. Different rows in the tables are the corresponding subjects, for which the model is built. Each column corresponds to an imposing user, who tries to passively attack the corresponding model. From this set of results we observe that for the majority of victim-attacker pairs we get 0% FAR. However, there are specific pairs that generate FAR up to 21.7% and 24.7% in each of the case studies. In addition, there exists specific subjects (like subject 7 in the shoes study) that have worse FAR in their models compared to the others. This suggests that there may be easier and harder to target subjects. Based on this observation, we conduct further experiments in Sec. 5.5.6 that evaluate what consequences there may be in the gait-based authentication system when an adversary can identify the best target and attempt mimicking the victim’s gait.

5.5.4 Expt. 2: Evaluation of Parameters that Affect Gait

In this set of experiments we evaluate how the walking speed and fatigue level of the subject will affect the authentication performance. The first experiment focuses on the walking speed. For each of the two case studies, we build auto-encoder and classification models based on slow
Figure 5.8: Per-subject leave-one-out performance

(a) Results of the shoe study

(b) Results of the socks study

97
walking data and test the performance of the models against fast walking data. This is repeated with fast walking data being the training set and slow walking data being the test set. Data from both morning and afternoon sessions are used for training on the shoes study in this experiment. The average EER and its standard deviation are reported in Fig. 5.9a for the socks study and Fig. 5.9b (first two error-bars, S-F) for the shoes study, respectively. From the results we have the observations that for the raw-stacked model with softmax, the average EER is 4.56% for the socks study with a standard deviation of 6.55%, and the average EER is 6.79% for the shoes study with a standard deviation of 18.01%. SVM seems to perform much worse in this experiment as for the socks study the average EER is 12.22% with a standard deviation of 13.73% and for the shoes study, the average EER is 25.79% with a standard deviation of 6.35%. Compared with the observations from Expt. 1, the results from this experiment indicate that when a gait authentication model is trained only on one pace, the performance drops significantly compared to training the models with data that contain multiple walking speeds. Specifically, the models that are trained on slow gait and tested with fast gait samples, and vise versa, have generated statistically significant differences in the EER scores when compared to the EER scores from models trained on both slow and fast gait samples. The generated p-value for the EER scores of the raw-stacked models with SVM between the two experiments is very small, i.e. $1.0023 \times 10^{-148}$ for the socks data and $3.8557 \times 10^{-129}$ for the shoes data. This concludes that the walking pace can greatly affect the authentication performance, and this is verified in both the socks and shoes studies. To reduce this performance degradation, the dataset that is used for training the user’s model, needs to include multiple walking speeds. The more significant performance degradation in the shoes study can be attributed to the higher sensitivity of the smart shoes used for this study and the fact that subjects in this study are asked to wear the same pair of shoes. A more detailed explanation of this is given in Section 5.5.7. Next, we discuss how the fatigue level of the subject will affect the authentication performance.

Fatigue from daily activities is considered to be a factor that can affect gait. To study the effect of reduced energy levels in the afternoon of the day, we perform a similar experiment
Figure 5.9: A summary of the results from all testing scenarios in Table 5.1 to the previous one, with the difference being that the training data are from the morning session and the testing data are from the afternoon session in the shoes study. This is then repeated by taking afternoon session data as the training set and the morning session data as the testing set. The results are summarized in Fig. 5.9b. From the figure, we have the observation that from the raw-stacked with softmax model, the Average EER and standard deviations are 7.11% and 11.42%, respectively. Comparing to the first set of experiments focusing on the walking speed only, the EER increases by 6%. This indicates that fatigue does play an important role in gait-based authentication. In addition, slightly different sensor placement when the subjects wear the shoes for the afternoon experiment may be another factor for this performance change. To reduce performance degradation from such cases, the dataset used for training the user’s model needs to include recordings from a set of subjects that have recorded their gaits in multiple different time-points of the day. By doing so, we can capture the day-to-day variance and variance from different sensor placement in shoe wearing.

To better estimate the effect of fatigue, we perform two more experiments by restricting the training and testing datasets to one walking pace, and the results are summarized in Fig. 5.9b. More precisely, the first experiment uses training and testing datasets from the slow pace morning (SM) and slow pace afternoon (SA) sessions, and the second experiment uses training and testing datasets from the fast morning (FM) and fast afternoon (FA) sessions. A comparison of the EER
<table>
<thead>
<tr>
<th>Expt.</th>
<th>Test scenario</th>
<th>Sensing</th>
<th>Metric</th>
<th>Classifier</th>
<th>$\mu$ (%)</th>
<th>$\sigma$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Slow vs Fast (S-F)</td>
<td>Shoes</td>
<td>EER</td>
<td>softmax</td>
<td>6.79</td>
<td>18.01</td>
</tr>
<tr>
<td>2</td>
<td>Slow vs Fast (S-F)</td>
<td>Socks</td>
<td>EER</td>
<td>softmax</td>
<td>4.56</td>
<td>6.55</td>
</tr>
<tr>
<td>2</td>
<td>Morning vs Afternoon (M-A)</td>
<td>Shoes</td>
<td>EER</td>
<td>softmax</td>
<td>7.11</td>
<td>11.42</td>
</tr>
<tr>
<td>2</td>
<td>Slow M vs Slow A (SM-SA)</td>
<td>Shoes</td>
<td>EER</td>
<td>softmax</td>
<td>9.03</td>
<td>9.56</td>
</tr>
<tr>
<td>2</td>
<td>Fast M vs Fast A (FM-FA)</td>
<td>Shoes</td>
<td>EER</td>
<td>softmax</td>
<td>12.73</td>
<td>16.60</td>
</tr>
</tbody>
</table>

Table 5.3: A summary of the experimental results on the performance of the raw-stacked model. M and A refer to Morning and Afternoon recording sessions, respectively. $\mu$ and $\sigma$ are the mean and standard deviation of the reported metric, respectively.

Although the performance is similar, it can be observed that the average EER is slightly better when only the fast walking pace data is used. This may be explained by the fact that fast walking can generate gait patterns that are more consistent since there is limited time in each gait cycle for deviations in the movement.

5.5.5 Expt. 3: Evaluation on the Impact of Training Time

The amount of data used for training a model can have a big impact on the performance. It is thus important to quantify how much training data a corresponding user needs to provide. In order to quantify that, we perform a similar set of experiments to those in Expt. 1 (Sec. 5.5.3), but in each iteration a different number of gait cycles is used to train the corresponding model. For a selected number of gait cycles, $c$, we pick $c$ consecutive gait cycles for training and the rest are used for testing. For each value in $c$, the experiment is repeated at most 10 times, if there are enough different sets of $c$ consecutive gait cycles. Average FAR results are summarized in Fig. 5.10. Fig. 5.10a reports the average results across all subjects, while Fig. 5.10b reports the results from individual subjects. Overall, we observe that FAR increases when the number of gait cycles used for training is reduced. In addition, from Fig. 5.10b we can observe that the performance depends on the subject as well. Some subjects seem to have their FAR fairly unchanged even when the smallest, i.e. 7, number of gait cycles is used for training. On the other hand, there are subjects that would benefit a lot from increasing their provided number of
gait cycles for training their models. By also taking into account the performance changes in gait from parameters such as walking speed and time of the day, it is advised to new enrolled subjects to provide not only increased number of gait cycles for training, but also diverse samples in terms of walking speed and collected time.

![Average FAR vs. number of gait cycles](image1.png)

![FAR of each subject vs. number of gait cycles](image2.png)

Figure 5.10: The impact of training time on the authentication performance

### 5.5.6 Expt. 4: Active Attacks through Gait Mimicking

In this experiment, a human subject is asked to perform gait mimicking to evaluate the robustness of our method. Under this scenario, impostors try to mimic the gait patterns of their victims by generating similar mechanical body movement as their victim do. This scenario assumes that the attacker has compromised the system’s database and can use their gait patterns against the models belonging to other subjects in the database. By doing that, they are able to identify which victim’s model gives the highest FAR. Once a victim is identified, the attackers are able to observe their victim’s walking patterns visually, and then mimic the victim’s gait patterns in order to increase their FAR even further.

In order to collect data for this gait mimicking scenario, we first identify candidate pairs of attacker-victim whose FAR is the highest compared to others. For this experiment, subject 9 was selected to be the victim, and subject 6 to be the attacker (impostor). Since some subjects are
Table 5.4 summarizes the gait mimicking results under different models. The FAR performance on the raw-stacked model with the passive gait leave-one-out (Expt. 1) is 12.7%. After performing gait mimicking, FAR on the raw-stacked model is increased to 15.18%. Although it seems that the impostor is able to increase their chance of being accepted, the differences in the FAR scores between the passive and mimicking attacks are not statistically significant (p-value is 0.268).

### 5.5.7 Summary and Discussion of the Experiment Results

We now summarize our experimental results and findings. First, we observe that the proposed methods can be successfully applied for gait-based authentication. Specifically, it is possible to achieve very low EER and FAR, based on early fusion of data from the two modalities. This indicates that gait data do not require complex and higher order features to improve the authentication performance. Based on the results from the first experiment (Fig. 5.7) we observe that the
average CV EER reaches as low as 0.01% for the smart socks platform and 0.16% for the smart shoes platform. To the best of our knowledge, this result is the best among all the related work, e.g., a 0.8% EER was reported in [103]. In addition, the leave-one-out FAR ranges from 0.54%, when using the smart socks to 1.96%, when using the smart shoes. Related studies have reported FAR performance of 3% with 11 subjects [109] and 6% with 32 subjects [110]. This indicates that our approach outperforms the works in the literature with similar subject populations.

Another observation is the different authentication performance between the two case studies. In general the socks study yields lower EER and FAR compared to that of the shoes study. This results may be attributed to multiple reasons. First, in the socks study, participating subjects are allowed to wear their own shoes. This indicates that wearing different shoes will contribute in differentiating the gait patterns, and thus make authentication easier in the socks study. In addition, differences in the subjects population and the sensing technology itself may also contribute to the different authentication performance. Nevertheless, this difference may give a hint to understand the extend to which shoe types affect gait-based authentication. Finally, differences in the GCF sensing technology between the two sensing platforms may also play a role in the performance differences.

From both Expt. 2 and Expt. 3, we observe that there are multiple parameters that can affect the gait-based authentication performance. Walking parameters such as speed and time of the day may have a significant impact on the gait patterns. In addition, the amount of gait cycles used for training the corresponding models may greatly affect the performance as well. Based on all these observations, it is recommended that when users provide their training data at the enrollment phase, they should provide longer and more diverse gait examples in terms of both walking speed and collection time of the day.
5.6 Conclusion

Gait-based authentication has recently gained great attention in the research community since it has shown promising results towards bridging the gap between usability and effectiveness of authentication methods. In this chapter, we present our approach to improve the robustness of gait-based biometric authentication with the introduction of multimodal learning. With the use of commercially available smart socks and medical-grade research prototype of smart shoes as our sensing platforms, we jointly collect GCF and ACC data that are then passed through a pipeline of analytic methods for segmentation, feature extraction and classification. The use of early fusion on the sensing data with autoencoders for feature extraction, and SVM for classification is shown to be a very promising design that can capture the specific characteristics of each modality, correlate them and achieve superior performance compared to the methods only using a single modality.
Chapter 6

Conclusions and Future Work

Studying normal and abnormal human gait can help in developing improved methods to assist with the new challenges associated with the rise of demand for gait rehabilitation and security in mobile devices, with biometrics. The ever-growing demand for gait rehabilitation requires gait analysis to be objective, automatic and scalable. To address these challenges, this dissertation first presents a data-driven approach for real-time gait phase detection to facilitate gait analysis and rehabilitation. The approach combines an infinite Gaussian mixture model (IGMM) and a parallel particle filter to classify the gait phases and update the model parameters. To further extend this work and provide better diagnostic tools for physical therapists, we will work towards designing better gait indices based on the extracted gait phases. These indices will help in objectively monitoring possible gait abnormalities that the physical therapists would not be able to identify through visual observation, only during the rehabilitative training. We also plan to extend the proposed data-driven gait phase detection algorithm to be an adaptive approach in which the number of particles will be determined in the runtime in a dynamic manner to achieve a better balance between the estimation accuracy and computational efficiency. Finally, more research needs to be conducted to achieve robust gait phase detection, so that sudden jumps and unstable predictions can be smoothed out with extra levels of filtering.

To further support objective gait diagnosis, we also design an integrative framework for gait
disorder diagnosis and advance smart gait rehabilitation. Gait features were developed for different categories including gait phases, mobility, balance and strength. MTFL, an advanced classification method, is used to train the different classification tasks that can classify the subject’s gait patterns. Data from PD and post-stroke patients, along with healthy subjects are used to evaluate the proposed methods. The proposed gait features successfully capture the underlying properties of each disease. MTFL is able to construct accurate classifiers and select the most important gait parameters for each classification task, ignoring the rest. The selected features capture the characteristics of each disease as described in the literature. As future work, we intend to provide more comprehensive gait disorder diagnostic tools for more complex gait disorders that are difficult for the clinicians to detect. We plan to assist their assessment process in the clinic, evaluate these analytic systems with properly designed clinical studies, and design new methods for rehabilitation progress evaluation and treatment plan development.

Finally, this dissertation presents a multimodal gait based authentication method that can be used in mobile devices. The proposed approach can improve the corresponding state of art and has been shown to be robust regardless of the selected sensing platform, or gait parameters that can affect gait patterns. There are still many open questions that need to be addressed before gait-based authentication systems can be adopted for everyday use on our personal devices. Biometric systems are vulnerable to different types of attacks, e.g. impersonation, replay, and spoofing. [66] gives an overview of all the possible vulnerable points in a generic biometric authentication system, including sensing devices, feature extraction modules, matchers, databases, and all communication channels connecting them. Impersonation attacks on gait-based biometric authentication systems can be a real threat to the security of the system and are very hard to model, since the sophistication level of an attacker and the resources available to them can vary. We have evaluated our proposed algorithm for gait-based biometric authentication with impersonation attack form a single subject. It will be our future work to further improve these vulnerable points with increased amount of human participant studies to enhance the robustness evaluation of the proposed gait-based authentication system. In addition, we will design more
user friendly wearable devices that can facilitate gait-based authentication with sophisticated sensing, but also improve usability, such as smart shoes that perform energy harvesting from the movements to increase battery life. Finally, advanced active learning methods will be used to help users decide if more diverse training set is required to improve the robustness of their authentication method.
Bibliography


Journal of Orthopaedic Translation, 5:26–37, April 2016. [1]


[28] Joel A DeLisa. Gait analysis in the science of rehabilitation, volume 2. Diane Publishing, 1998. [1.1.1] [1.1.1] [2.2.1] [2.3.1] [4.1.1] [4.1.2] [4.1.3] [4.1.4] [4.1.5] [4.3.3] [4.3.3] [4.3.3] [4.3.3]


[34] Simona Ferrante, Emilia Ambrosini, Paola Ravelli, Eleonora Guanziroli, Franco Molteni, Giancarlo Ferrigno, and Alessandra Pedrocchi. A biofeedback cycling training to improve locomotion: a case series study based on gait pattern classification of 153 chronic stroke patients. Journal of neuroengineering and rehabilitation, 8(1):1, 2011. 1.1.1, 2.3.2


[55] Kyoungchul Kong, Joonbum Bae, and Masayoshi Tomizuka. Detection of abnormalities
in a human gait using smart shoes. In Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2008, volume 6932, page 69322G. International Society for Optics and Photonics, April 2008. 2.4


[58] Sang-Hong Lee and Joon S Lim. Parkinson’s disease classification using gait characteristics and wavelet-based feature extraction. Expert Systems with Applications, 39(8):7338–7344, 2012. 1.1.1, 2.3.2


[63] Johannes Michalak, Nikolaus F Troje, Julia Fischer, Patrick Vollmar, Thomas Heidenre-


[69] Sara Mulroy, JoAnne Gronley, Walt Weiss, Craig Newsam, and Jacquelin Perry. Use of cluster analysis for gait pattern classification of patients in the early and late recovery phases following stroke. Gait & posture, 18(1):114–125, 2003. 1.1.1 2.3.2


[79] Shijia Pan, Tong Yu, Mostafa Mirshekari, Jonathon Fagert, Amelie Bonde, Ole J Meng-


[86] Kara K Patterson, William H Gage, Dina Brooks, Sandra E Black, and William E McIlroy. Evaluation of gait symmetry after stroke: a comparison of current methods and...


[96] Pete B Shull, Kristen L Lurie, Mark R Cutkosky, and Thor F Besier. Training multi-parameter gaits to reduce the knee adduction moment with data-driven models and haptic


