Examining the impact of Normalization and Footwear on Gait Biometrics Recognition using the Ground Reaction Force

by

James Eric Mason
Bachelor of Software Engineering, University of Victoria, 2009

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

MASTER OF APPLIED SCIENCE

in the Department of Electrical and Computer Engineering

Supervisory Committee

Examining the impact of Normalization and Footwear on Gait Biometrics Recognition using the Ground Reaction Force

by

James Eric Mason
Bachelor of Software Engineering, University of Victoria, 2009

Supervisory Committee

Dr. Issa Traoré, (Department of Electrical and Computer Engineering)
Supervisor

Dr. Hong-Chuan Yang, (Department of Electrical and Computer Engineering)
Departmental Member

Abstract

Supervisory Committee

Dr. Issa Traoré, (Department of Electrical and Computer Engineering)
Supervisor

Dr. Hong-Chuan Yang, (Department of Electrical and Computer Engineering)
Departmental Member

Behavioural biometrics are unique non-physical human characteristics that can be used to distinguish one person from another. One such characteristic, which belongs to the Gait Biometric, is the footstep Ground Reaction Force (GRF), the temporal signature of the force exerted by the ground back on the foot through the course of a footstep. This is a biometric for which the computational power required for practical applications in a security setting has only recently become available. In spite of this, there are still barriers to deployment in a practical setting, including large research gaps concerning the effect of footwear and stepping speed on footstep GRF-based person recognition. In this thesis we devised an experiment to address these research gaps, while also expanding upon the biometric system research presented in previous GRF recognition studies.

To assess the effect of footwear on recognition performance we proposed the analysis of a dataset containing samples for two different types of running shoes. While, with regards to stepping speed, we set out to demonstrate that normalizing for step duration will mitigate speed variation biases and improve GRF recognition performance; this included the development of two novel machine learning-based temporal normalization techniques: Localized Least Squares Regression (LLSR) and Localized Least Squares

Regression with Dynamic Time Warping (LLSRDTW). Moreover, building upon previous research, biometric system analysis was done over four feature extractors, seven normalizers, and five different classifiers, allowing us to indirectly compare the GRF recognition results for biometric system configurations that had never before been directly compared.

The results achieved for the aforementioned experiment were generally in line with our initial assumptions. Comparing biometrics systems trained and tested with the same footwear against those trained and tested with different footwear, we found an average decrease in recognition performance of about 50%. While, performing LLSRDTW step duration normalization on the data led to a 14-15% improvement in recognition performance over its non-normalized equivalent in our two most stable feature spaces. Examining our biometric system configurations we found that a Wavelet Packet Decomposition-based feature extractor produced our best feature space results with an EER average of about 2.6%, while the Linear Discriminant Analysis (LDA) classifier performed best of the classifiers, about 19% better than any of the others. Finally, while not the intended purpose of our research, the work in this thesis was presented such that it may form a foundation upon which future classification problems could be approached in a wide range of alternative domains.

Table of Contents

List of Tables

List of Figures

Acknowledgements

The research presented in this thesis would not have been possible were it not for valuable assistance and research direction provided by my supervisor, Dr. Issa Traoré. Although this thesis went far beyond what is typically required of a Master's thesis and faced numerous delays, Dr. Traoré had confidence in my abilities and supported me through times when it looked as if this work would never reach its conclusion. Without Dr. Traoré, I never would have been introduced to proper research methodology and my new found passion for machine learning.

I would also like to thank the team over at Plantiga for initiating this research, including CEO Quin Sandler, who worked hard to track down the data used to accomplish my research objectives. I would further like to express my gratitude to Jennifer Baltich with the Human Performance Laboratory at the University of Calgary, who provided the data samples used throughout this thesis.

Additionally, I would like to thank my external examiner, Dr. Yvonne Coady, and department committee member, Dr. Hong-Chuan Yang, who took time out of their busy schedules to review my work and provided excellent feedback in the process.

Finally, I am sincerely grateful to my wife Pairin and family for all the support they have given me over the years. I realize these last few years have been tough for you as I have tried to balance working a full time job, completing my research, and spending time as a family, but you were always understanding, believed in me, and kept me rounded.

Chapter 1

Introduction

Over the past several decades national security concerns and the need to deter increasingly sophisticated fraudsters have driven demand for a new generation of reliable person identification tools. Traditional identification technologies have been built around *something a person has* (such as an identification card) or *something a person knows* (such as a password), but, to improve reliability, newer technologies are increasingly including *something a person is*, the physical and behavioural characteristics that define an individual. As technology continues to improve, the automatic recognition of a person based on physical or behavioural characteristics, referred to as biometric recognition, seems destined to have a profound impact on physical and cyber security while we progress through the 21st century.

1.1 Context

The first automated biometric system was a fingerprint identification tool developed in the 1970s. This tool, called AFIS (Automated Fingerprint Identification System), was used to assist with forensics investigations of criminal activities. Prior to the mid-1990s biometric devices were typically bulky and expensive, making them difficult to deploy; but with the recent rapid expansion in computing power it has become much easier to deploy biometric systems. The decreasing cost and size of biometric devices has now made it practical to install them for instant identification at everyday access points, whereas, formerly, these devices were reserved for law enforcement or high security environments. However, while technology has enabled wider use of biometrics, it has also made it easier to circumvent them.

Well known biometrics based on physical characteristics, including fingerprints, facial features, and iris patterns, have shown vulnerabilities to spoofing attacks. The paper, "Biometric attack vectors and defences" [1], by Chris Roberts, referenced a number of successful attacks targeting physical biometrics over the past 15 years. It was discovered that fake fingerprints made from gelatine, and taken from enrolled persons, were able to fool optical fingerprint devices with false acceptance rates as high as 68-100%. Even more alarming, one team of researchers discovered a technique to successfully "lift" residual fingerprints from scanners using graphite powder, tape, and enhanced digital photography; opening the possibility for easy access to sensitive biometric data. Meanwhile, facial recognition has been found vulnerable to spoofing attacks that

involved playing back images of a person's face. And iris scans have also been successfully spoofed, using high resolution photographs of an enrolee's iris.

To address the potential for spoofing, Roberts suggested several techniques, with a primary focus on increasing complexity of the data collection process and capturing proof that an incoming data came from a living person. Such techniques include: requiring blinking, randomization of fingers asked for during a fingerprint scan, thermal measurements, and surface reflectivity among others. There is another category of biometrics for which a living person is often considered an implicit part of data. This category of biometrics is known as behavioural biometrics, and refers to the measurable characteristics of a person's actions. The strength of these biometrics comes from their dynamic nature (the relative ease of requiring variability during identification) and complexity required to reproduce, thus spoof, the actions observed. Such biometrics commonly include speech recognition, keystroke dynamics, and walking gait. Although recognition performance by behavioural biometrics is typically weaker than physical biometrics, this category of biometrics presents a major advantage regarding user acceptance, as they are often seen by people to be less intrusive than physical biometrics [2].

1.2 Problem Statement and Research Objectives

The ever increasing use of biometrics to enhance traditional security devices has come under increased scrutiny in recent years. Privacy advocates often make the argument that biometrics present a high risk in the case of a compromise, as they cannot simply be reset like more traditional identification mechanisms. Researchers have demonstrated that today's biometric tools may not necessarily be as secure as we might imagine. While many end users have shown resistance to the intrusive nature of biometric collection techniques, particularly those involving captured images. Biometrics structured around unique behavioural, rather than physical, characteristics have been suggested as a means to provide enhanced security with less risk and greater convenience to end users. One such biometric factor that has attracted a lot of attention over the past 10 years is the human gait.

Gait biometrics refers to the unique aspects of human locomotion that can be captured and used for recognition purposes. Much of the recent research into gait biometrics has focused on extracting features from gait sequences captured on video. However, this technique raises similar concerns to those of physical biometrics over both intrusiveness and potential for forgery (via video playback attack). An alternative gait biometric technique that may be less objectionable and perhaps even more secure, involves extracting unique walking features from the force signatures generated as a person steps over floor plate sensors or sensor-loaded shoes. This footstep-based technique offers several potential advantages over the video approach: it does not require the capture of intrusive images; it is less susceptible to interference from obstructions (i.e. changes in

lighting or objects obstructing the view); and its interface requires a complicated transfer of force over a short period of time that, with today's technology, would be very difficult to reproduce in a spoofing attack. Nevertheless, this technique is still young and has only been studied by a small number of researchers [3].

Previous attempts at performing footstep recognition have generally focused on comparing the recognition ability of well-known classifiers (the models that determine the likelihood of an identification match) and comparing the discriminative properties of footstep feature extraction approaches. Unfortunately, there is not yet any standard publically available footstep force signature datasets, and the studies behind these attempts used different datasets of varying quality, making it difficult to accurately compare the effectiveness of their chosen methods [4]. Moreover, large research gaps remain regarding both the effect of data normalization on classification success and that of shoe type variation on recognition performance.

The force metric examined by this thesis is the ground reaction force (GRF), a measure of the force exerted by the ground back on the foot during a footstep. The primary objective of the research presented in this thesis is to address the large research gaps regarding the effect of normalization and shoe type variation on footstep GRF-based recognition. A secondary objective is to expand upon the work of previous researchers with respect to the processes of feature extraction and classification.

Preliminary research suggests variations in shoe type will have a negative impact on recognition performance [5], and that a relationship exists between stepping speed and force amplitude that could possibly be used to improve recognition performance via normalization [6]. The experiment proposed in this thesis aims to verify both assertions. Furthermore, as previously noted, much of the existing footstep GRF recognition research has been devoted to feature extraction and classification techniques. It must also be noted that these techniques were not tested on a single dataset but rather on a different dataset for each study, making inter-study comparison difficult. The experiment proposed in this paper aims to compare some of the various techniques, together with a previously untested technique, on a single, high quality dataset to better assess their effectiveness. It is hoped that the research presented in this thesis will make a significant contribution to the present day understanding of footstep GRF-based recognition and pave the way for deployment in a real world system.

1.3 Summary of Contributions

Contributions of this research can be described in the following points:

Feature Extraction

The research presented in this thesis contributes to the present day knowledge base for GRF feature extraction in two ways. Unlike previous studies, that extracted feature sets from data obtained using at most three GRF sensors, this study extracts a feature set from data obtained using eight GRF sensors. Additionally, the research presented in this thesis compares the feature extraction techniques, applied by previous studies across different datasets, on a single dataset to more accurately assess their relative effectiveness.

Normalization

There has been little-to-no previous research into the effects of feature set normalization on footstep GRF recognition. The research presented in this thesis contributes to the present day knowledge base by providing a detailed analysis of normalization based on stepping speed. No known previous research has provided such an analysis with regards to the impact of stepping speed as a means to normalize footstep GRF features. We introduce a novel regression-based approach to stepping speed-based feature set normalization and compare it with the amplitude-based normalization [3] and stepping speed-based resampling normalization [7] techniques used in previous footstep GRF recognition studies.

Classification

Existing studies have deployed some of the strongest known classification techniques to perform footstep GRF recognition. The work presented in this thesis compares the best of these techniques using features obtained from the novel, normalized, eight-sensor feature set discussed in the previous two research contributions. The research presented in this thesis also contributes to the present day knowledge base by performing classification using a classification technique that has not yet been used by any other footstep GRF study.

Shoe Variation

The final contribution that this thesis makes to present day research relates to variation in shoe type, which might be expected to affect a system performing footstep GRF recognition. To date, only a single study [5] has attempted to assess the impact of differences between shoe types used to train a recognition system and those used to authenticate with a recognition system. The research presented here expands on that study, performing a detailed analysis of recognition results obtained from a dataset containing three different shoe types.

1.4 Thesis Outline

The remaining chapters of this thesis are structured as follows:

Chapter 2

This chapter describes the field of gait biometrics and provides a historical overview of work that has been done in the field to date. It goes on to explain where the footstep GRF fits into the field of gait biometrics, and reviews the footstep GRF recognition literature that forms the basis for the research presented in later chapters.

Chapter 3

This chapter presents the experimental setup and introduces the methodology used to achieve the thesis objectives. It covers the selection of a development dataset containing data of a single shoe type and proposes a biometric system composed of feature extractors, normalizers, and classifiers to perform GRF-based person recognition.

Chapter 4

This chapter compares four different feature extraction techniques previously used for GRF-based recognition in other studies. Theoretical background is provided for each feature extractor together with a discussion of each implementation. Preliminary GRF recognition results are acquired using the development dataset and presented for the parameter optimization of each extractor.

Chapter 5

This chapter demonstrates the performance of various normalization techniques on the extracted feature spaces from chapter 4. Two novel normalization techniques are introduced here and theoretical background is provided for these and several other well-known existing techniques that are also examined. To determine the effectiveness of normalization the results from applying these normalization techniques are compared with the non-normalized results of the previous chapter.

Chapter 6

This chapter presents the theoretical background and implementation for five different classifiers that were selected for analysis in this thesis. Each classifier is tuned across the best-performing feature spaces acquired from development dataset in chapters 4 and 5. Finally, the feature extractor-normalizer-classifier combinations that achieved the best results are summarized for comparison with the results over the evaluation dataset in chapter 7.

Chapter 7

This chapter demonstrates the results obtained after applying the best footstep GRF-recognition systems, outlined in chapter 6, to an evaluation dataset containing previously unseen data samples with different shoe types.

Chapter 8

This chapter discusses the findings behind the GRF footstep recognition experiment. The effects of various techniques are compared, with practical implications and explanations for possible sources of error presented. Finally, the chapter concludes by examining techniques that could potentially be used to improve upon the results discovered in this thesis.

Chapter 9

This chapter provides a final summary of the research presented in previous chapters. The major findings are highlighted and remaining problems and areas for future work are discussed.

Chapter 2

Background and Related Work

The GRF, in the field of biometrics, is defined as the force of the ground pushing back on a person's foot while the foot is in contact with the ground. This force is equal and opposite to the force exerted by the foot on the ground and varies during motions like walking. The GRF is part of a greater study of human locomotion, referred to as gait biometrics. To better understand where the GRF fits into the overall field of gait biometrics, this chapter presents an overview of the field and identifies relevant research that has been done to date. The chapter concludes by examining research specific to the GRF biometric, and identifies the research gaps that inspired the novel research presented later in this thesis.

2.1 Authentication using the Gait Biometric

The gait biometric is among the most recent biometric traits to be studied for use in human recognition systems, with the first studies beginning in the early 1990s [8]. Gait is classified as a behavioural rather than physical biometric. Traditionally, biometrics based on unique physical traits, such as fingerprints, have been the focus of biometric recognition studies; however, with recent technology improvements we have begun to realize that certain aspects of our behaviour, like gait, may also be sufficient for recognition purposes.

Biometric recognition using gait presents a number of advantages over traditional biometric traits: it is generally considered unobtrusive, as it can be measured in way that does not require a person to alter his or her typical behaviour; it does not require a person to present any more information than is already available to a casual observer; and studies have suggested it is very difficult to imitate [9]. In his research, Cattin [5] makes special mention of the ability of the gait biometric to perform a living person test. The test is described as the ability to determine whether the owner of a trait being observed is alive and physically present or not. Traditional biometrics, such as fingerprints, often fail this test as the traits observed can be faked with present day technology. The security of the gait biometric lies in the incredible difficulty required to spoof it, thus the living person test is considered intrinsic to the method.

There are a variety of characteristics that define the human gait and a variety of techniques used to extract these gait characteristics. In a summary of research in the field

of gait biometrics, Derawi et al. [10] suggest there are 24 different components that, together, can uniquely identify an individual's gait. The components examined and data extracted are largely restricted by the instrumentation used for measurement. The approaches used to accomplish gait recognition relate directly to the instrumentation needed to extract gait data, and fall into three categories: the machine vision approach, the wearable sensor approach, and the force plate approach. The following subsections examine these approaches and describe the research that has been done in each. The studies mentioned in these sections measure performance according to two modes of operation: verification and identification. In the identification mode performance is measured by the rate at which an identity can correctly be assigned to a data sample, while in verification mode performance is measured using the verification Equal Error Rate (EER), a measure of the error incurred when matching an identity to a given data sample.

2.1.1 The Machine Vision Approach

The machine vision (MV) approach to gait biometrics involves capturing gait information from a distance using video recorder technology. This is the most common approach to gait biometric recognition referenced in current literature [9], having benefited from the availability of large public datasets such as the NIST MV gait database [8]. Recognition via MV has been accomplished using two different techniques: model-free and model-based recognition algorithms. The model-free technique, often referred to as the silhouette-based technique, involves deriving a human silhouette by separating out a

moving person from the static background in each video frame. Using this technique, classifiers are developed around the observed motion of the silhouette. The less commonly used model-based technique involves imposing a model onto human movement [11]; this is often accomplished by extracting features, such as limbs and joints, from captured images and mapping them onto the structural components of human models for recognition [12].

Over the past decade gait recognition using MV has been attempted by a number of researchers using a variety of methods with promising results. In 2003, Wang et al. [13] developed a silhouette-based technique that used the feature space dimensionality reducing Principal Component Analysis (PCA) together with the Nearest Neighbour and Euclidean Nearest Neighbour classification algorithms. This approach achieved identification rates in the 70-90% range, varying on the dataset and acceptance criteria used. In 2005, Boulgouris et al. [14] proposed a novel silhouette-based system for gait recognition using Linear Time Normalization (LTN) on gait cycles. The system demonstrated an 8-20% improvement in its identification rates when compared with existing methodologies at the time. Another study that same year by Lu et al. [15] achieved an identification rate of 92.5% using a Genetic Fuzzy Support Vector Machine (GFSVM) classifier; this result improved upon the results of the Nearest Neighbour and standard Support Vector Machine (SVM) tested against the same dataset.

In 2006 Cheng et al. [12] introduced a gait recognition system that used a Hidden Markov Model (HMM) and, unlike previous systems, was designed to perform

recognition on subjects walking down different paths. It accomplished this by first recognizing the walking direction, then applying an appropriate identifier to the determined path; identification rates achieved by this system varied in the 80-90% range across differing datasets and acceptance criteria. Another silhouette-based study in 2006, by Liu and Sarkar [16], used a generic population HMM (pHMM) to normalize gait dynamics, then used PCA to reduce the feature space and a Linear Discriminant Analysis (LDA) classifier to perform classification; the use of a HMM for normalization was unique to this study and demonstrated how normalization could be used to improve upon recognition performance. In 2009 a study by Venkat and De Wilde [17] took a different approach and attempted to reduce the computational intensity of silhouette-based recognition techniques by examining sub-gaits, defined as smaller localized frames, rather than entire gait images. This technique yielded an identification rate range of about 75-90% across various datasets and acceptance criteria. However, when vision or motion obstructing factors such as carrying condition and, particularly, clothing condition were considered the identification rate dropped to as low as 29%.

Few researchers to date have studied the effects of the various factors that can obstruct human gait recognition; however, a real world gait recognition system would most likely need to be designed to handle such events. To address the issue of gait variability and obstructions, also known as covariate factors, Bouchrika and Nixon [18] proposed a model-based system to extract human joint positions and model gait motion using elliptic Fourier descriptors. Their 2006 study successfully extracted 92.5% of the heel strikes observed in a dataset containing both visible joints and joints occluded by clothing,

demonstrating that key features of motion analysis could still be tracked in the presence of covariate factors. A follow-up study in 2008 [19] examined the effects of footwear, clothing, carrying condition and walking speed, and, using the model-based approach to joint extraction together with a K Nearest Neighbour (KNN) classifier, achieved an identification rate of 73.4% against a database containing variations of these covariate factors.

Two further studies attempted to mitigate the weaknesses of the MV approach to gait recognition by fusing it with a secondary biometric factor. In 2002, Cattin [5] developed a system that fused the data from a video sensor recognition system with a force plate footstep recognition system to recognize an individual walking in a monitored room. The results of his study were promising with a verification EER of 1.6%. In 2006, Zhou and Bhanu [20] developed a different multifactor biometric technique that combined an MV gait recognition system with a facial recognition system. This system had the benefit of only requiring a single sensor for capturing video and achieved an identification rate of 91.3%.

All studies discussed so far have dealt with recognition using images captured from a video sensor, one unique study, which, however, was not based on visual data but best falls under the MV category, proposed audio-based footstep recognition. In 2006, Itai and Yasukawa [21] took a wavelet transform technique, widely used in feature extraction for speech recognition, and applied it to feature extraction for audible footsteps. Using this

technique an identification rate of 80% was achieved, suggesting this could be an exciting new realm for study in the field of gait recognition.

2.1.2 The Wearable Sensor Approach

Biometric recognition using wearable sensors (WS) is a new approach to gait biometrics that aims to use sensors attached to the human body to perform recognition. Much of the early research into gait-aware wearable sensors came from medical studies that focused on their usefulness for detecting pathological conditions [22]. Research into the usefulness of the WS approach for biometric recognition has been, at least partly, held back due to the lack of large publically available datasets [10]. However, the WS approach to biometrics presents a number of advantages, including the ability to perform continuous authentication, which would not always be possible with sensors fixed to a physical location. Over the past 10 years a number of studies, using a variety of techniques, have investigated the feasibility of the WS approach.

In 2006, Gafurov et al. [23] developed a WS biometric recognition system using an accelerometer sensor attached to the lower leg. This study first collected test subject data then uploaded it to a computer, rather than performing real time classification. The experiment achieved a verification EER of 5% when recognizing individuals using a histogram similarity classification technique. Another study in 2006, by Huang et al. [24], presented a recognition system based on sensors embedded in a shoe. The sensors used included a pressure sensor, tilt angle sensor, gyroscope, bend sensor and

accelerometer. This system was developed to transmit gait data from the shoe to a computer in real time, and used the PCA feature reduction technique together with a SVM classifier to accomplish a 98% identification rate on a small sample dataset. A follow-up study by Huang et al. [25] in 2007 applied a Cascade Neural Network with a Node-Decoupled Extended Kalman Filtering (CNN-NDEKF) classifier to the shoe-based WS system and achieved a 97% identification rate.

One major weakness of the WS approach is the potential inconvenience or discomfort that may be caused by attaching sensors to the human body. For this reason WS research has tended to focus on one of two unobtrusive WS techniques: shoe-based monitoring techniques, like [24] and [25] described in the previous paragraph, and phone-based monitoring techniques. Both techniques make use of equipment that is already a part of daily life and require no alterations to typical behaviour. In the last few years the increasing computational power and wider use of smart phones has sparked a number of studies into feasibility of phone-based monitoring techniques.

A study in 2009, by Spranger and Zazula [26], described a biometric recognition system that worked with a feature set consisting of cumulants of accelerometer data captured by a mobile phone attached to a person's hip. The system achieved a 93.1% identification rate on a small dataset using PCA for feature dimensionality reduction and a SVM classifier. A separate study in 2009 by Fitzgerald [27] demonstrated a system, designed for possible future use in mobile phones, that worked with accelerometer and gyroscope data captured by a Nintendo Wii controller. Gait cycles captured by the system were

normalized with respect to time. User recognition for the system was examined using KNN, Naive Bayes and Quadratic Discriminate Analysis (QDA) classifiers, with the KNN classifier performing best, achieving an identification accuracy of about 95%. In 2010, Derawi et al. [28] collected data from accelerometers attached to a belt on the legs of 60 volunteers, generating a much larger dataset than used in the other WS studies described in this chapter. This study focused on cycle length as a metric and, using a Cross Cyclical Rotation Metric (CRM), achieved a verification EER of 5.7% for person recognition. Although the device used by the study was not a mobile phone, the application of this system for use in mobile phones was noted as an important area for future research. In 2011 Nickel et al. used a HMM classifier on accelerometer data from commercially available mobile phones to perform person recognition and achieved a verification EER of about 10%. The study worked with a relatively large dataset of 48 subjects and was particularly promising for the field of WS-based authentication, because it proved that even a standard, commercially available mobile phone could now be used for gait recognition purposes.

2.1.3 The Floor Sensor Approach

The floor sensor (FS) approach to gait biometrics involves recognizing people based on the signals they generate as they walk over sensor-monitored flooring. Data captured by floor sensors typically falls into two categories: binary image frames of the foot while it is in contact with the ground, and single dimensional force distribution plots, which describe the force exerted by the foot over time. Most FS technology was developed for

the study of biomechanical processes; particularly for improving performance in athletics and discovering the effects of pathological conditions such as diabetes [29]. The first studies using FS technology for gait recognition began in the late 1990s [3]. Over the past 10 years a small but increasing number of studies have examined the FS approach to gait biometrics. Some, like [5] by Cattin, described in machine vision section of this thesis, combined the FS recognition approach with another gait recognition approach, such as the MV approach, to improve recognition accuracy; however, most have focused on using the FS approach for single factor recognition.

Open research into using footsteps as a biometrics dates back to a 1997 study by Addlesee et al. [30]. In this study, load cell floor sensors were used to capture partial GRF data for 15 volunteers and, using a HMM classifier, a 91% footstep identification rate was achieved. Three years later, Orr and Abowd [31] outfitted a floor tile with a set of force sensors to capture the GRF profile for 15 volunteers. In the study, ten features were extracted and normalized, then passed to a Euclidean Nearest Neighbour (ENN) classifier for recognition; the result was a 93% identification rate. Another study in 2005, by Suutala and Röning [32] used a floor sensor called ElectroMechanical Film (EMFi) to capture the GRF for ten volunteers. The primary focus of this study was to compare various classifiers, combine various classifiers, and examine the effects of rejecting unreliable data samples from classifier training. The study found that the SVM and Multilayer Perceptron Neural Network (MLP) classifiers performed best; the strongest corresponding identification accuracy on their most complicated dataset was around 92%, which increased to 95% when the most unreliable 9% of sample set was rejected. A later

study in 2007 by Moustakis et al. [7] captured the GRF for a larger dataset of 40 volunteers. This study used a feature extraction technique built on Wavelet Packet Decomposition (WPD) to detect the transient characteristics and distinguishing features of the GRF, then applied a SVM classifier to the feature set; the result was a 98.3% identification rate.

In 2009 Ye et al. [33] presented a unique technique for FS-based gait biometrics: instead of performing recognition on a footstep, like most previous studies, they developed a system that could recognize a person by upper-body movements performed while standing on a force plate. The study obtained the center of pressure for the foot, and monitored its movement as instructed actions were completed by a person on a force plate. The study achieved its best results using a Neural Network classifier, with a verification EER in the 1-12% range. Two other studies, one in 2008 [34] and another in 2011 [35], also took a different approach to FS-based gait biometrics, opting to perform gait recognition using binary images of footsteps, rather than GRF force signatures. The 2008 study by Suutala et al. [34] examined the shape and pattern of individual footstep image frames, as well as the displacement between feet during footsteps; it achieved a maximum identification rate of 84%, using a Gaussian Process classifier. The 2011 study, by Yun [35], focused primarily on subjects in bare feet and also extracted features from individual footstep frames together with footstep displacement. Using a MLP classifier, trained with the extracted features, the study achieved a 96% identification rate.

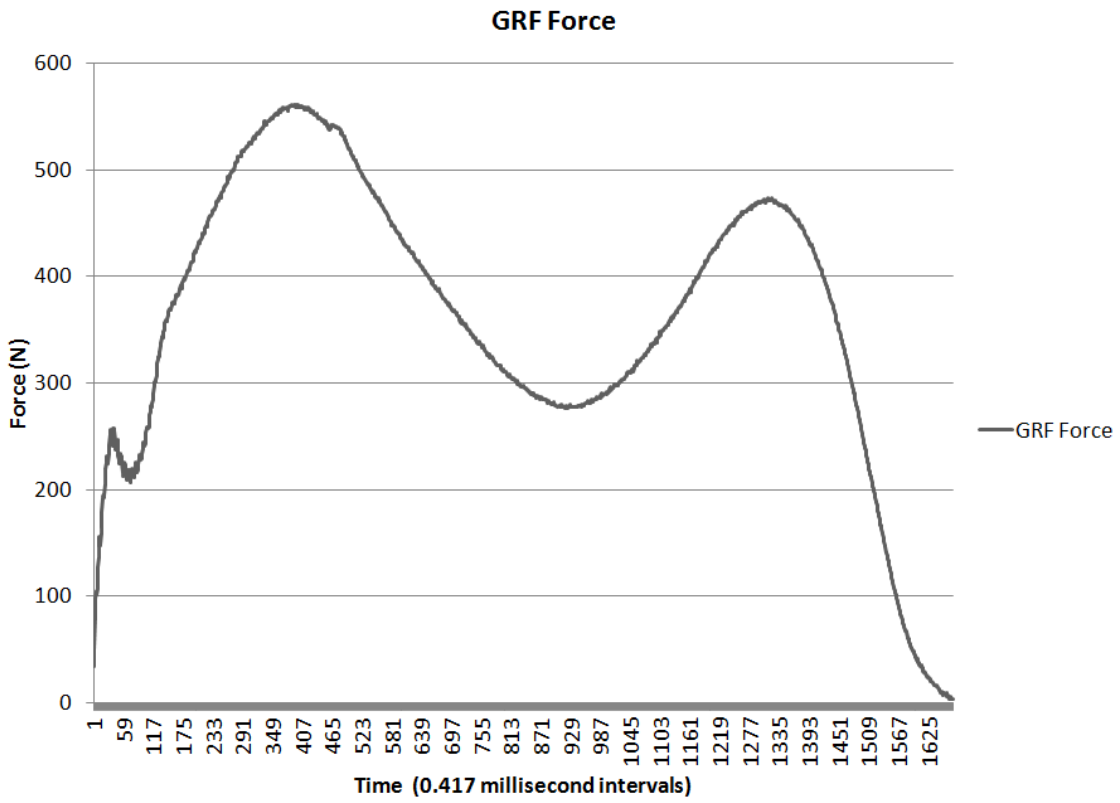

Figure 2.1: This graph demonstrates the force-time series representation of a footstep commonly used for FS-based recognition.

Research focused on the FS approach to gait biometrics, like the WS approach, has been disadvantaged due to the lack of any publically available datasets. The lack of a publically available dataset makes it difficult to compare results across studies. To address this issue, one group at the University of Wales Swansea set out to develop such a dataset. In 2007, Rodríguez et al. [3] published a study of a FS-based recognition system and introduced a dataset of footstep force signatures covering 41 persons and over 3000 footsteps; the intended purpose was to verify the data and make the dataset available at some future point. In their process, they presented a holistic and geometric feature set, and, using an SVM classifier achieved a verification EER of 11.5% for person

recognition. A later study [4] in 2008, dealt with a dataset expanded to 55 persons and more than 3500 footsteps. Using the same classifier as the previous study, but, additionally normalizing and optimizing the feature set, a verification EER of 13% was achieved; the small increase in error rate was attributed to the larger dataset. The database in the 2008 study was said to have been made publically available, but, at the time of writing, no longer appears on the project website [36]; nevertheless, the project web site has indicated a larger dataset is currently being packaged for future release.

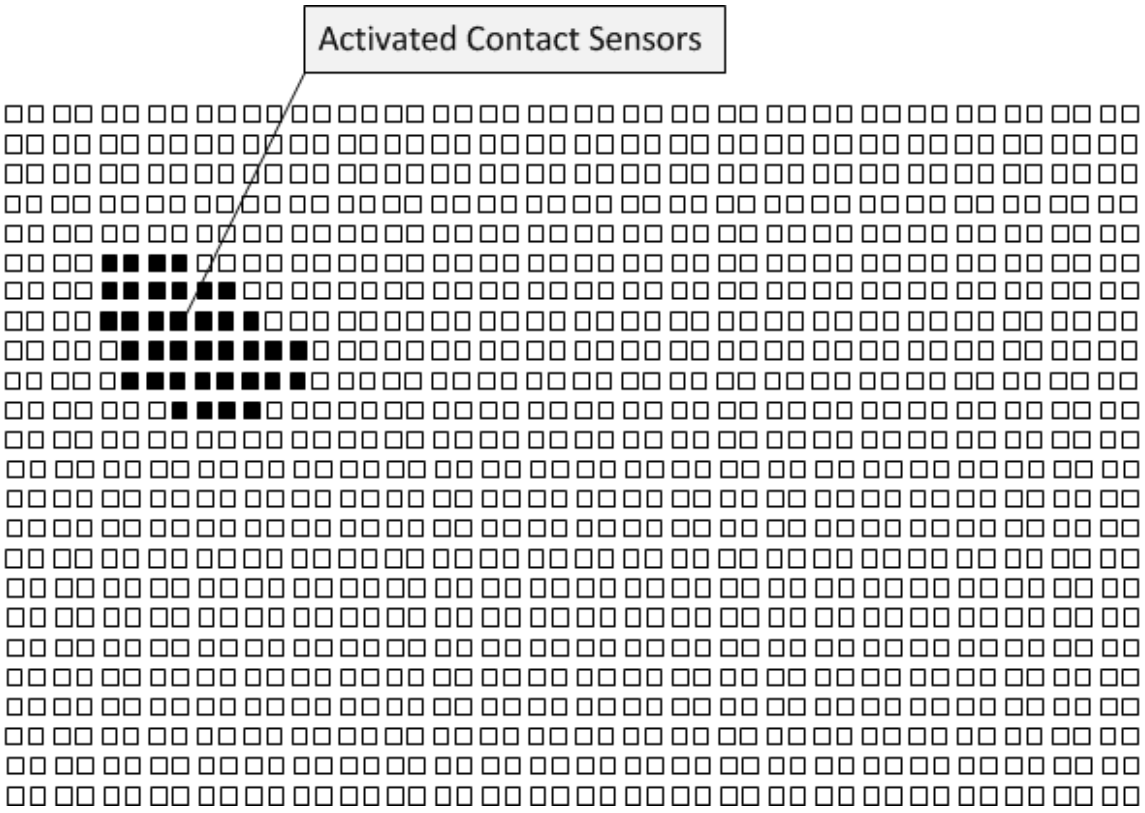

Figure 2.2: Binary Footstep Frame. This diagram demonstrates a single frame of a footstep on a pressure sensitive sensor array. The darker region represents locations where the footstep is detected as being in contact with the floor. The series of frames produced by a single footstep can be used for FS-based footstep recognition.

2.2 Recognition using the Footstep Ground Reaction Force

This section examines the components that make up the GRF and the research that has been put toward using footstep GRF for recognition purposes. The studies covered here form the foundation for the research presented in later chapters. The previous section demonstrated how gait biometrics can be categorized into three different approaches, two of these approaches can be used to capture the GRF: the GRF can be obtained using the FS approach with force plates, or, less commonly, using a shoe-based WS approach. At the moment most research has dealt with GRF captured via force plate sensors, but there are projects, including the work of Plantiga [37], that are examining incorporating GRF recognition into a shoe-based wearable sensor. Research demonstrated in this thesis deals specifically with GRF data collected via force plate.

The footstep GRF, shown according to the Kistler force plate coordinate system in figure 2.6, is represented by a three component force vector, with each component reflecting a different aspect of the footstep. The vertical force component of the footstep, shown as Fz in figure 2.6, represents the vertical acceleration of the body, and is larger when the body is accelerating upward and smaller when the body accelerates downward. The time series vertical GRF (Fz) of a single footstep is shown in figure 2.3; it has two distinct peaks that correspond first to the phase in the step where the foot impacts the ground, and then to the phase where the foot pushes up off the ground. The anterior-posterior force, shown as Fy in figure 2.6, represents the horizontal friction between the foot and the ground. This component, shown in the time series in figure 2.4, is largely responsible for horizontal motion and its peaks and troughs correspond to forward acceleration and impact

breaking, respectively. Lastly, the medial-lateral component of the footstep, shown as Fx in figure 2.6, represents friction forces perpendicularly to the direction of motion; these forces, shown in the time series in figure 2.5, reflect the rotation of the ankle during a footstep. Most researchers have tended to focus on the vertical component of the GRF for recognition; however, studies have indicated the anterior-posterior and medial-lateral forces contain valuable subject specific information [5].

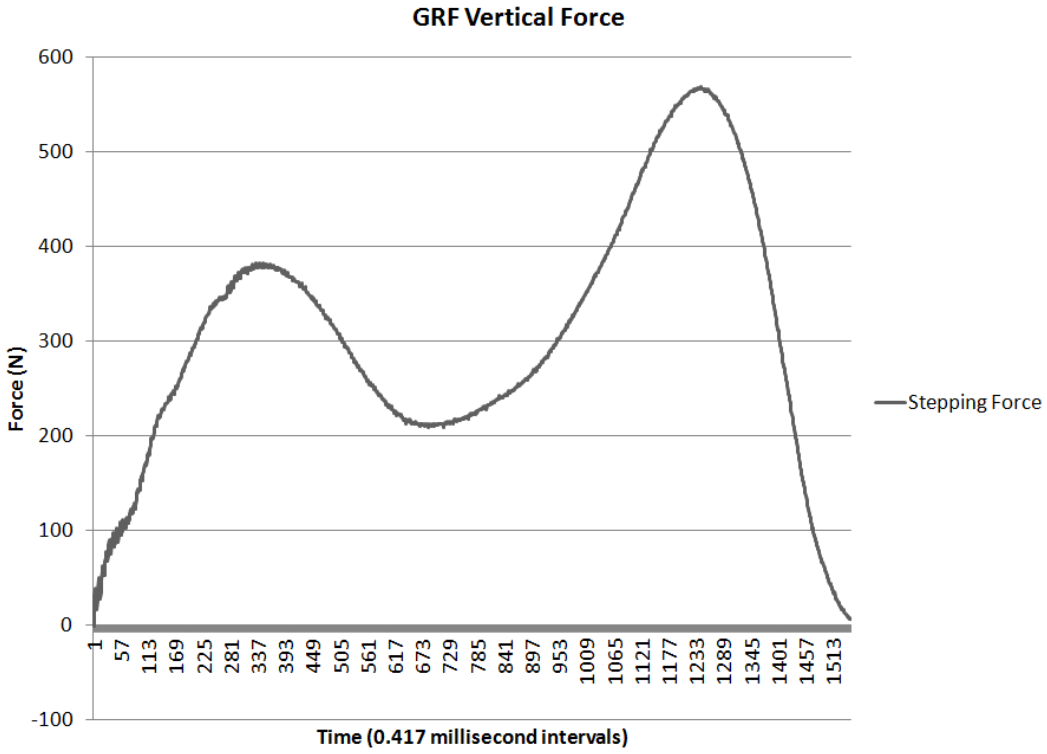

Figure 2.3: This figure demonstrates the GRF vertical force for a footstep.

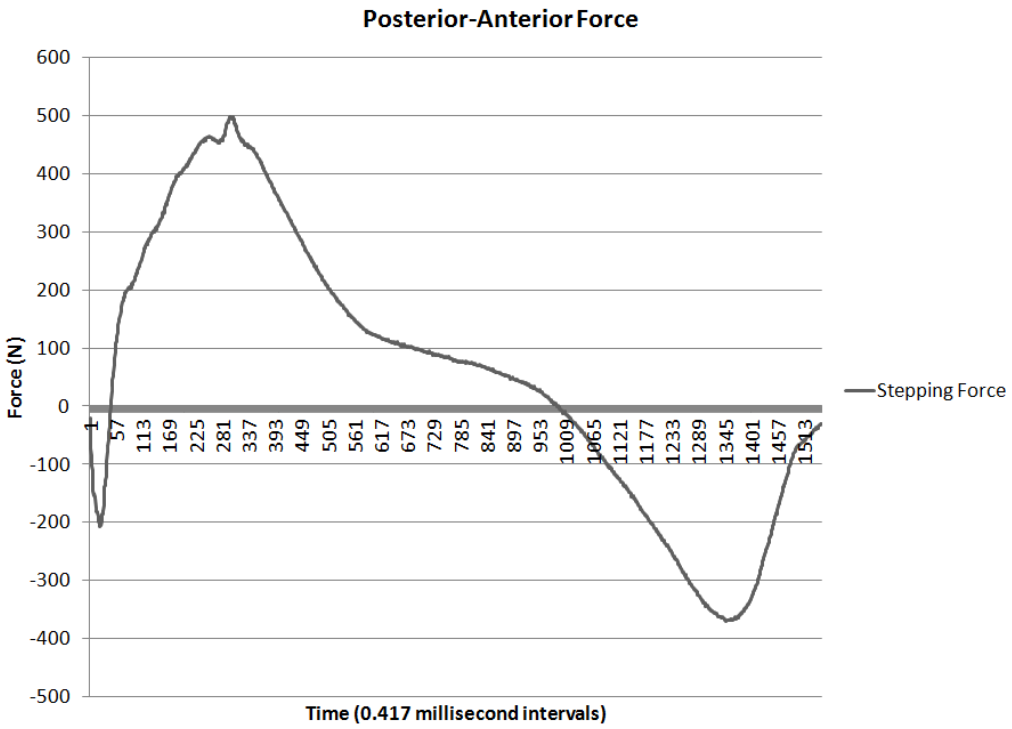

Figure 2.4: This figure demonstrates the GRF posterior-anterior force for a footstep.

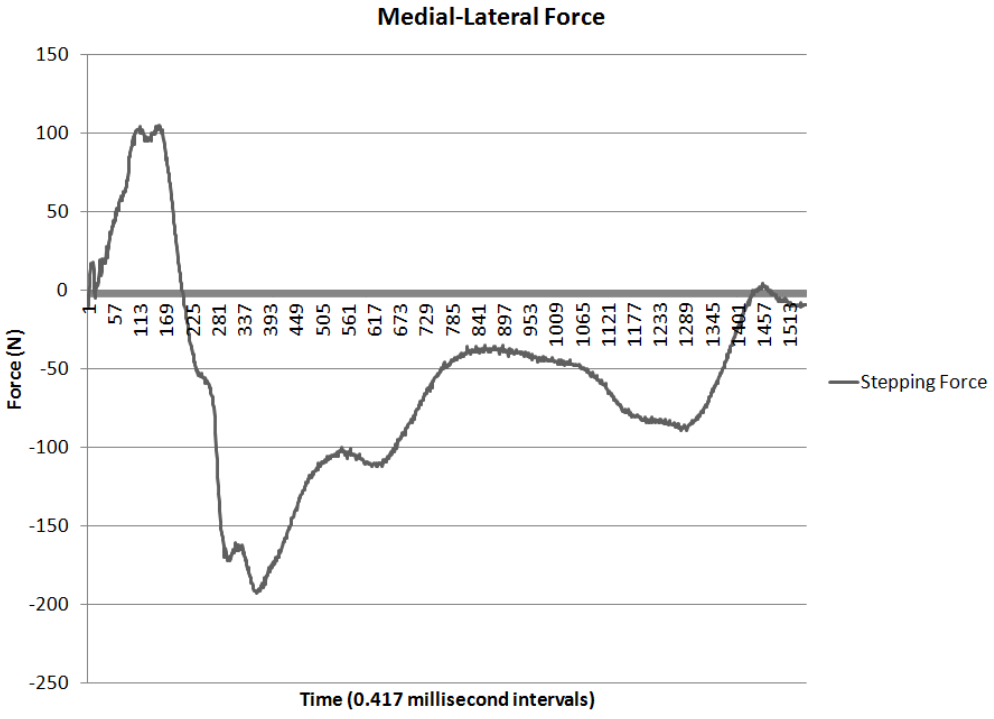

Figure 2.5: This figure demonstrates the GRF medial-lateral force for a footstep.

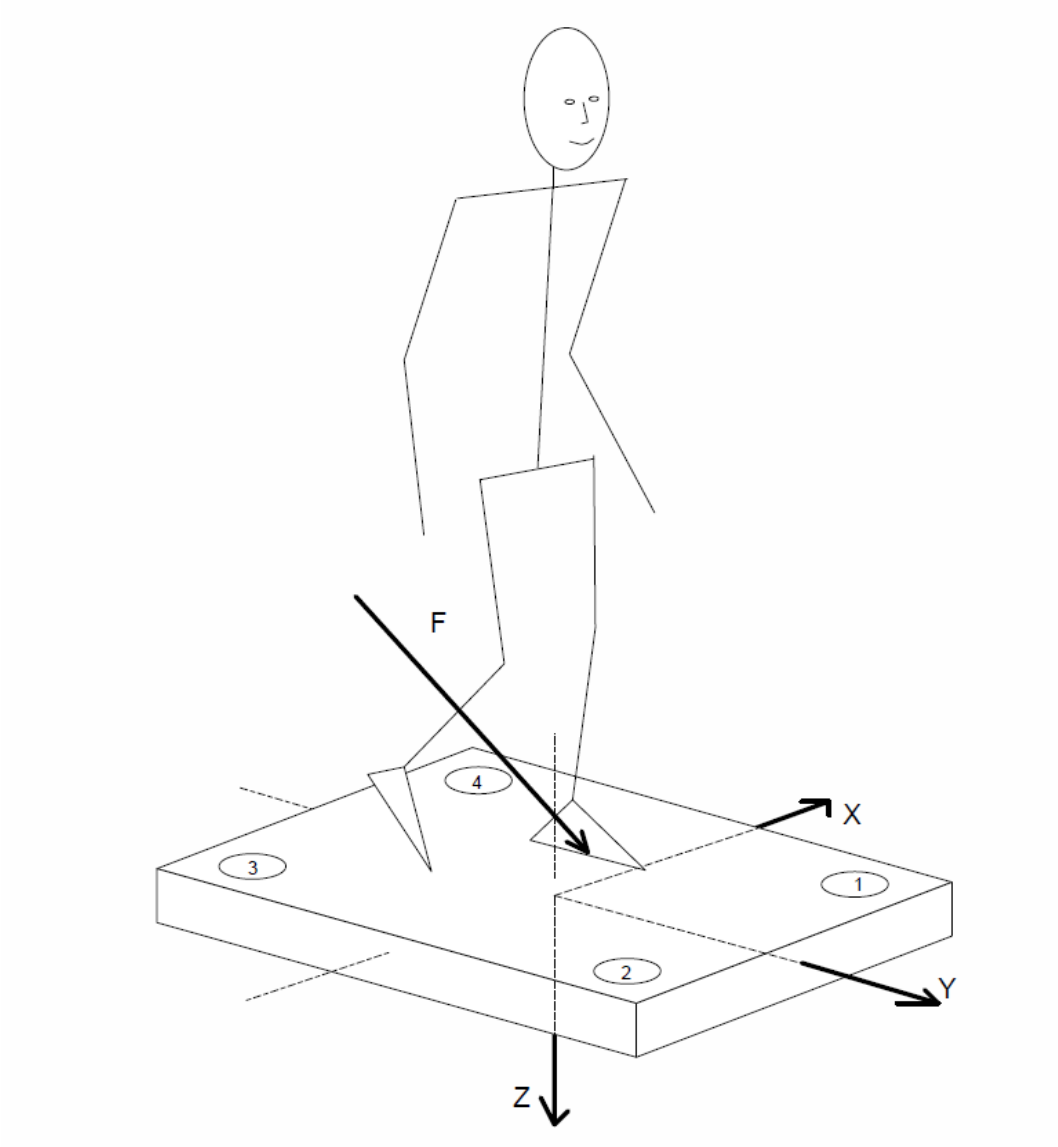

Figure 2.6: Kistler Force Plate Coordinate System [38]. The force labelled 'F' represents the stepping force vector. The force plate translates this into its vertical (Z), anterior-posterior (Y), and medial-lateral (X) components.

Since the footstep GRF was first proposed as a biometric in 1997 [4], only a small number of researchers have examined it for its stand-alone recognition capability. The results of some of the key GRF recognition studies are shown in table 2.1 below. Most of these results were previously discussed in section 2.1.3, but this table makes it possible to

compare the studies in areas relevant to this thesis. One caveat in comparing different studies is that most use different datasets, and factors like the size and quality of the dataset can have a significant impact on performance. It must also be noted that the results demonstrated often come only from the dataset used for development, and, when evaluation datasets are used, performance tends to decrease. This decreases in performance was demonstrated by Rodríguez et .al in [3] and [4], with an increase in verification EER of 21% and 330% respectively when evaluation datasets were used instead of development sets. Furthermore, while some studies measured performance as the ability of a classifier to identify a person using footstep profiles, others were based on the ability of a system to verify a person's identity given credentials and a footstep.

Group / Year	Database	Classifier	Results	Multiple Shoes Tested	Normalization	Training Samples / Person
Addlesee et al. / 1997 [30]	300 steps / 15 persons	HMM	ID rate: 91%	No	No	10 steps
Orr and Abowd / 2000 [31]	1680 steps / 15 persons	NN	ID rate: 93%	Yes	Yes	10 steps
Cattin / 2002 [5]	480 steps / 16 persons	Euclidean Distance	Verif EER: 9.4%	Yes	No	6 step cycles
Suutala and Röning / 2005 [32]	440 steps / 11 persons	SVM	ID rate: 94%	No	No	27 steps
Moustakidis et al. / 2007 [7]	2800 steps / 40 persons	GK-SVM	ID rate: 98.3%	No	Yes	7 steps
Rodríguez et al. / 2007 [3]	3174 steps / 41 persons	SVM	Dev Verif EER: 9.5%	Yes	Yes	40 steps
Mostayed et al. / 2008 [39]	18 steps / 6 persons	Histogram Similarity	Verif EER: 3.33 ~ 16%	No	No	1 step
Rodríguez et al. / 2008 [4]	3550 steps / 55 persons	SVM	Dev Verif EER: 3%	Yes	Yes	40 steps

Table 2.1: This table compares different approaches to GRF footstep recognition. Results refer to those obtained using a development dataset. Step cycles refer to the combination of the right and left footsteps that make up a walking cycle.

From the results shown in table 2.1, it seems that the GRF is capable of producing similar, or perhaps better, recognition performance than achieved using the MV-based recognition approach. While GRF features appear to be less susceptible to covariate factors than gait features captured by video [5], there still is a potential for factors like varying shoe type or stepping speed to reduce GRF recognition accuracy. Of the 8 studies examined, only 4 attempted recognition using datasets that included multiple shoe types for a single person. Likewise, normalization, a natural technique to reduce the impact of variance such as disparities in measured stepping speed, was also only applied in 4 studies. The following subsections expand upon the discussion of how GRF research to date has addressed the two primary emphases of this thesis, shoe type variance and stepping speed normalization, as well as the important secondary emphases: features and classification approaches.

2.2.1 Feature Extraction

The first step in building a biometric recognition system involves identifying the most discriminative features that can be extracted from raw data. Ideally, features used for recognition should appear consistently for all persons tested, yet show enough variance such that there is no overlap in feature space between two or more persons. In reality, finding such features can be a difficult task, particularly for behavioural biometrics, and there is usually at least some degree of overlap in feature spaces. In the studies referenced in table 2.1, four different types of features are presented: geometric features [31, 7, 32,

3, 4], holistic features [3, 4, 39, 30], spectral features [5, 32], and wavelet transform features [7].

Geometric features refer to a feature set that is determined using well-recognized geometric attributes like max and min points, as well as statistical attributes like mean and standard deviation. Most studies into GRF recognition have used geometric features as either a primary feature set for classification, or as a comparison feature set to assess the effectiveness of an alternative feature extraction technique. In [31], a biometric recognition system was built using the mean, standard deviation, area under the curve, and extrema (min/max) points for a footstep GRF graph; it was noted in the study that the mean and standard deviation appeared to show the highest discriminative power. An attempt to optimize the geometric feature set was made in [4]. In this study the geometric feature set for the footstep GRF graph contained extrema points, the distances between extrema points, the area under the curve, the norm, the mean, the length, and the standard deviation. To determine the best features an exhaustive search was performed, searching for the combination of features that minimized the verification EER on a development dataset. The result was a reduction in feature dimensionality from 42 features to 17. The optimal features included 11 extrema point features, 2 area features, 2 norm features, and 2 standard deviation features, for which a 27% increase in performance was noted. While the optimized geometric features showed a significant improvement in performance, [3, 4] and [7] demonstrated comparatively better results with different feature extraction approaches, and [32] found that combining the geometric feature set with a set from another feature type produced a significant increase in performance.

Holistic features have appeared as a promising alternative to geometric features in several GRF recognition studies. The holistic approach to feature extraction generally involves dividing the raw data into some arbitrary number of equally spaced points (or samples) then allowing the most discriminative points to reveal themselves to a classifier. The simplest technique, used in [30], involves passing the data directly into a classifier. This approach can run into problems, as it tends to lead to incredibly large feature space dimensionality, but there are solutions. In [39] the dataset was simplified using representative histogram. Yet, the most effective simplification technique was demonstrated in [3], where the raw holistic feature set originally contained 4200 features but was simplified using the PCA approach. The use of PCA made it possible to generate a reduction in dimensionality while retaining as much information about the original data as possible. Applying PCA demonstrated that the first 80 principal components contained 96% of the original information, while reducing dimensionality by 98%. When performance of these 80 holistic features was compared against the performance of a geometric feature set, a 46% increase in performance was observed.

Another alternative to the geometric feature set was proposed in [7]. This approach performed feature extraction by first translating GRF data to the time-frequency domain using a wavelet packet transform, then extracting the 100 most discriminative features from the data using a form of optimized wavelet packet decomposition. Converting data to the time-frequency domain allowed the complex information and patterns associated with the GRF to be represented in a simpler form. This characteristic proved very advantageous for classifying footstep GRF samples with walking speeds that differed

from those the system was trained on; in these instances the increase in performance, compared with the recognition results from a 16 feature standard geometric set, was as high as 66%. When training and testing data came from the same walking speed range the increase in performance over the alternative geometric feature set was also around 66%.

Additionally, spectral features, features extracted from the frequency domain, have proven useful in two previous GRF recognition studies. In [5], a feature set, derived from the windowed Power Spectral Density (PSD) function of the derivative GRF, was suggested to provide stronger recognition ability than could be achieved with a geometric feature set. The PSD function is particularly useful because it shows the strength of energy variations as a function of frequency. By identifying the frequencies at which variations are strong, this function could make it easier for a classifier to identify the most important features. To capture a low dimensionality feature set from the PSD function a novel Generalized PCA (GPCA) technique was used, and it was found that the first 10 Principle Components contained more than 90% of the dataset variance. The technique produced a reasonably strong verification EER of 9.4%, however, no equivalent geometric feature set was tested on the dataset, so it was not possible to make a direct comparison between the two techniques. In [32], two holistic feature sets, derived from the frequency domain of the GRF and its derivative function, were combined with a geometric feature set. While, on their own, these spectral feature sets performed worse than the geometric set in this study, when all three sets were combined the resulting performance was 36% better than the best standalone result.

Not only did the studies in table 2.1 vary in their approaches to feature extraction, but they also varied in the components and quality of data collected. In [31], [32], [3] and [4], an averaged GRF was obtained and examined, rather than one split into its three vector components. In [5], [30], and [39], only the vertical component of the GRF was used for final classification analysis; both [5] and [39] regarded it as the best discriminator. The only study that used all three components of the GRF for classification was [7]. No study examined GRF classification for more information rich data samples involving the output from 4 or more GRF component output signals, opening the possibility for further study into GRF feature extraction using higher quality data. Furthermore, since previous studies did not share the same dataset, a better relative comparison of feature set performance could be achieved by comparing the feature extraction techniques of different studies on the same dataset.

2.2.2 Normalization

It is very difficult for a person to perform the same action twice with no measureable difference between the two attempts. When collecting a feature set for the footstep GRF, differences in walking speed can have a large impact on the timing and amplitude of the extracted features. Unfortunately walking speed, and by extension stepping speed, appear to be extremely difficult to regulate with high precision, even in a controlled experiment. One technique that can be used to account for natural variation in data is normalization. Normalization assumes that some sort of relationship exists between two or more

variables, and, by using this relationship, variables can be projected onto the same reference point for a more accurate comparison.

Of the studies referenced in table 2.1, only 4 applied normalization to their dataset; two of these were from the same research group [3, 4]. None of the studies covered their chosen normalization techniques in detail, and it appeared that only simple normalization techniques were used. In [31], data normalization was mentioned, but no detail was given regarding the technique used or the target of the normalization. In [7], data was normalized around the weight of test subjects so, when loads of 5% and 10% of the subjects body weight were added during testing, it was possible to adjust the feature set to a common weight reference point. The study also used a simple resampling-based Linear Time Normalization (LTN) technique to address differences in step sample length (duration); however, the feature extraction technique also focused on capturing features less sensitive to walking speed variation and no non-normalized results were presented for reference. Finally, in [3] and [4], feature sets were normalized with respect to the absolute maximum value of the GRF footstep profile; this simple approach would appear to account for variations in stepping force but not step duration.

While no study dealing directly with GRF for recognition examined the actual effects of using normalization to address differences in walking speed, there is evidence from other related studies that suggests such an approach may achieve better recognition results. A study examining gait using the MV approach found that applying LTN to feature data improved identification performance over non-normalized feature sets by 8-20% [14];

this result implied the existence of an identifiable relationship between walking speed and observable gait characteristics. The impact of walking speed on GRF was also examined in [6] as part of a human kinetics study that analyzed its relationship with the vertical GRF component. The study examined the difference in the amplitude of the vertical GRF across three different walking speeds for 20 volunteers and found: the maximum amplitude increased by 2% when walking at a normal speed compared to a slow speed, it increased by 6% when walking at a fast speed compared to a normal speed, and by 9% when walking at a fast speed compared to a slow speed. The identification of such a clear relationship between walking speed and GRF supports the need for further investigation into utilizing this relationship to improve recognition results.

2.2.3 Classification Approaches

In biometric recognition, classifiers are the algorithms that take a feature set as input then attempt to either assign it an identity, or verify that it corresponds to a provided identity. Classifiers can be categorized according to two different models: generative models and discriminative models [40]. Generative classifiers involve first estimating an input distribution, then the modeling of class conditional densities, and finally calculating the posterior class probability via Bayes rule (this being the probability that a set of features corresponds to a given class); for instance, to learn the posterior class probability function P(X|Y), where X is a class and Y is a feature, a generative classifier would first need to estimate the a priori probability for each class P(X) and class conditional probability P(Y|X), then apply Bayes rule to get the intended result. Conversely, discriminative

classifiers are based on decision boundaries that minimize the classification error loss over the true class conditional probabilities and model posterior class probabilities directly or learn a direct map to class labels [41]; using the previous example, a discriminative classifier might attempt to determine P(X|Y) directly. Of these two approaches, discriminative classifiers have generally proven best for footstep recognition [32]. In the studies presented in table 2.1, 9 different classifiers were tested and the most successful methods were identified in the classifier column. In these studies, only 3 generative classifiers (Maximum Likelihood (ML), LDA, HMM) were attempted, while the remaining 6 were discriminative (KNN, SVM, MLP, Radial Basis Function (RBF), Learning Vector Quantization (LVQ), C4.5). In most studies a single classifier was trained to make decisions across the full feature space. However, in [32], three different instances of a chosen classifier were trained using three distinctive regions in the feature space, and the posterior probabilities returned by the three classifiers were fused into a single probability using combination rules; the result was a 46% decrease in error by the strongest classifier.

The most commonly used classifiers in the studies of table 2.1 were variants of the KNN classifier. The KNN classifier is a simple algorithm that assigns a feature set to the closest known identity (class), measured as the distance between a known feature set and the input feature set being classified. Variants of this classifier include the histogram similarity approach described in [39], and the Euclidean distance approach described in [5]. In [31], a relatively high identification rate of 93% was achieved using simple KNN,

but, in [7], [32], and [3], other classifiers were compared with KNN and showed better recognition performance.

After KNN, the next most widely used, and most successful GRF classifier in table 2.1, was the SVM classifier. The SVM classifier is a supervised learning method that constructs a hyperplane or set of hyperplanes in a high dimensional space, making the separation of complex classes easier. In [7], [3], and [32] this classifier generally demonstrated the strongest performance when compared against a number of other classifiers, with a performance increase ranging from 6% to 60% over the standard KNN classifier. However, in [7], LDA, a classification technique that searches for the linear combination of features to best separate two or more classes, demonstrated similar performance to the SVM classifier when large feature sets were tested. Also, in [32], a MLP classifier demonstrated only slightly weaker identification rates than that of the SVM classifier. None of the remaining classifiers covered by [32] and [7] (RBF [32, 7], LVQ [32], ML [7], C4.5 [7]) performed much better than the KNN classifier, while the HMM classifier, attempted in [30], has not appeared in more recent GRF recognition research.

Clearly the choice of classifier plays a strong role in GRF recognition performance, but classifiers must be trained and the number of samples used for training can also affect performance. In the studies of table 2.1, the number of samples used for training ranged from 1 [39] to 40 [3, 4] GRF samples per person. However, only [3] attempted to find an optimal number of footsteps for classifier training. In this study, recognition was

tested across 1 to 63 training steps and performance was demonstrated to increase substantially until about the 40th step, after which it levelled off. While 40 training steps appeared optimal for this particular study, it is important to note that, since each study used a different dataset, the optimal number of training samples for one study cannot be expected to be equivalent in another.

The number of footsteps used per single classification attempt is another factor that can affect recognition performance. Only two of the studies in table 2.1 examined multi-footstep classification. In [5], training and classification were done using two step cycles (the right and left steps that form a walking cycle). In [32], multi-footstep classification was compared directly with single footstep classification; a 76% increase in performance over single step classification was observed using 2 step classification, while a 95% increase in performance was observed with 4 step classification. The study also applied a sample-rejection strategy to ignore unreliable data samples from training and testing. Then, using 3 footstep classification, with the most unreliable 1% of the dataset rejected, the study achieved a 100% identification rate.

One final classification consideration regards best practices when demonstrating classification results. In [3] and [4], the separation of test data into a development and evaluation set was emphasized. When building a classifier, the development dataset is used to optimize the algorithm to the chosen feature set, while the evaluation set contains previously unseen data, and is used to confirm the results of the development set. Many of the studies demonstrated in table 2.1 did not use an evaluation dataset, so, for purpose

of making better comparisons, all results demonstrated in the table referred to those obtained using a development dataset. Furthermore, because footstep GRF recognition is such a new field of study, most research has been restricted to relatively small datasets compared to more traditional biometrics.

Classification algorithms fall into a broader field of machine learning, and have received extensive research over the past few decades. Recognition using the GRF has clearly benefited from the development of classifiers in related biometric research, and, it is apparent from the studies in table 2.1 that most of the strongest known classifiers have already been attempted by existing research. However, this area of research is constantly evolving and there is always room for testing previously untested classification algorithms for GRF recognition. Moreover, since most datasets previously used in GRF recognition were built on limited, low resolution sensors, it is possible that some algorithms may show an increase in performance and/or a lower training cost given a more descriptive dataset.

2.2.4 Shoe Type

Even with a highly discriminative normalized feature set and a strong classification algorithm, there will always be some level of variability in human gait that makes footstep GRF recognition difficult. One such source of variability can arise from the use of a different shoe for classifier training than was used for identification or verification. Unlike stepping speed variance, which can be calculated directly from a GRF step time

series, there is no way to determine that an individual is wearing a new shoe type based on the footstep GRF signature alone, therefore normalization by shoe type does not seem possible without further environment information. Of the 8 studies in table 2.1, only 4 examined a sample dataset containing more than one shoe type.

The effect of variable shoe type on classification performance is debatable. In [31], multiple shoe types were captured for classification and it was concluded that shoe type has little effect on the ability to perform footstep GRF identification. However, this study never indicated whether the multiple shoe types were used for the same person or simply across the whole test group, nor did it provide any information as to whether a different shoe type was used for training than for testing. Conversely, in [5], a more detailed analysis revealed that testing with shoe types unknown to the classifier could potentially have a very negative effect on recognition performance. Poor performance was demonstrated by a Euclidean distance classifier when new shoe types appeared in test data, however, when multiple shoe types were used for both training and testing there was a considerable improvement in performance. Finally, the remaining two studies, [3] and [4], both mentioned that two or more shoe types per person were included in their datasets, yet no analysis was done to study effects of these shoe types on classifier performance. Neither study had poor enough results to suggest that using multiple shoe types was having a very negative impact on classifier performance, though not enough information was provided to completely rule it out.

So, while a small group of researchers have studied footstep GRF recognition using datasets with multiple shoe types, only a single study, [5] by Cattin, went into any detail regarding the effect of shoe type on recognition. Moreover, even in [5], critical pieces of information were missing from analysis. For instance, Cattin found that classification performance was weaker in a multi shoe type dataset than in a single shoe type dataset; however, he did not specify whether the choice of shoe type altered stepping speed (a factor that could potentially be mitigated by normalization). Furthermore, Cattin only examined the effect of shoe type on classification using a Euclidean distance classifier. He discovered that training the classifier with multiple shoe types could increase recognition performance across a multi shoe type dataset, but that may not always be an option in a real world environment. It remains to be seen whether a stronger classification algorithm, different feature set, and stepping speed normalization could potentially mitigate the performance decrease that appears when performing classification on a footstep with a shoe type unknown to the classifier.

2.3 Summary

This chapter presented an overview of the field of biometric gait recognition and described where GRF footstep recognition fits into the field with respect to previous research. While the first half of this chapter provided a historical overview of research into gait recognition, the second half focused primarily on the footstep GRF biometric. Footstep recognition using the GRF was shown to be achievable using two of the three primary approaches to implementing gait recognition: the WS approach and the FS approach. However, all research, to date, appears to have been based around using the FS approach to perform GRF recognition.

In assessing research relevant to GRF recognition, 8 different studies were identified and compared according to the set of criteria in table 2.1. Upon review, research was found to be lacking with regards to footstep speed normalization and shoe type analysis, while the research areas of feature extraction and classification also left room for new experimentation. None of the listed studies attempted to assess the impact of normalizing the GRF features as a function of stepping speed. Only a single study examined the effect of shoe type on GRF recognition in any detail, leaving a lot of room for further analysis. Feature extraction and classification were more thoroughly covered by existing research, but could still benefit from the use of a more descriptive dataset, a better cross comparison of approaches, and the trial of a previously untested classifier type. The next chapter proposes a novel experiment to address the identified shortcomings and expand upon the work of previous footstep GRF recognition researchers.

Chapter 3

Experimental Design and Dataset

GRF datasets tend to be large and contain a high degree of variability, making potentially important patterns difficult to assess visually. Discovering a process that is able to isolate and exploit these patterns to best assist in distinguishing of one individual from others is at the core of GRF recognition-based research. Previous research has suggested that machine learning techniques, algorithms that specialize in the extraction and prediction of patterns, are well suited for accomplishing this objective. These techniques have been of particular interest in the areas of feature extraction and classification, but could also potentially be extended to be used in the normalization of data via regression analysis.

This chapter presents an experiment that applies some of the most powerful feature extraction, classification, and normalization algorithms available to address the difficulties in performing GRF recognition. The objective is to provide a more comprehensive comparison of previously used recognition techniques while also demonstrating novel methods to increase recognition performance. With respect to the demonstration of novel recognition methods, this chapter aims to establish an optimized framework to verify the correctness of the two problem statement assertions: that variation in shoe type between training and testing data can have a negative impact on recognition ability, and that a potentially useful relationship exists between stepping speed and the GRF force signature.

3.1 Experimental Design

The objectives of this work depend on the ability to accurately collect and compare performance metrics for a variety of GRF recognition techniques. These performance metrics can be obtained using a biometric system, a two stage system that consists of an enrolment and challenge phase. During the enrolment phase the system is provided with data to learn the biometric signatures of enrolled individuals. During the challenge phase new data samples are provided to the system and recognition is performed. Biometric systems can operate in one of two modes (described in section 2.1): verification mode or identification mode. For the purpose of this thesis, a GRF biometric system has been developed with all results acquired in verification mode. This novel experimental biometric system has been setup to allow for multiple configurations of feature extraction, normalization, and classification techniques. The relative strength of each biometric system configuration can be compared in verification mode according to the Equal Error Rate (EER) generated during each configuration's challenge phase. The EER identifies the rate at which a biometric system's False Acceptance Rate (FAR) is equal to its False Rejection Rate (FRR), when given some input dataset. This metric can be visually observed using a Detection Error Trade-off (DET) curve as shown in figure 3.1.

$$FRR = \frac{N_{FR}}{N_{SR}} \times 100 \qquad FAR = \frac{N_{FA}}{N_{SA}} \times 100$$

Equation 3.1: N_{SR} and N_{SA} refer to the total number of rejected and accepted samples by the biometric system, while N_{FR} and N_{FA} refer to the number of samples incorrectly rejected and accepted, respectively.

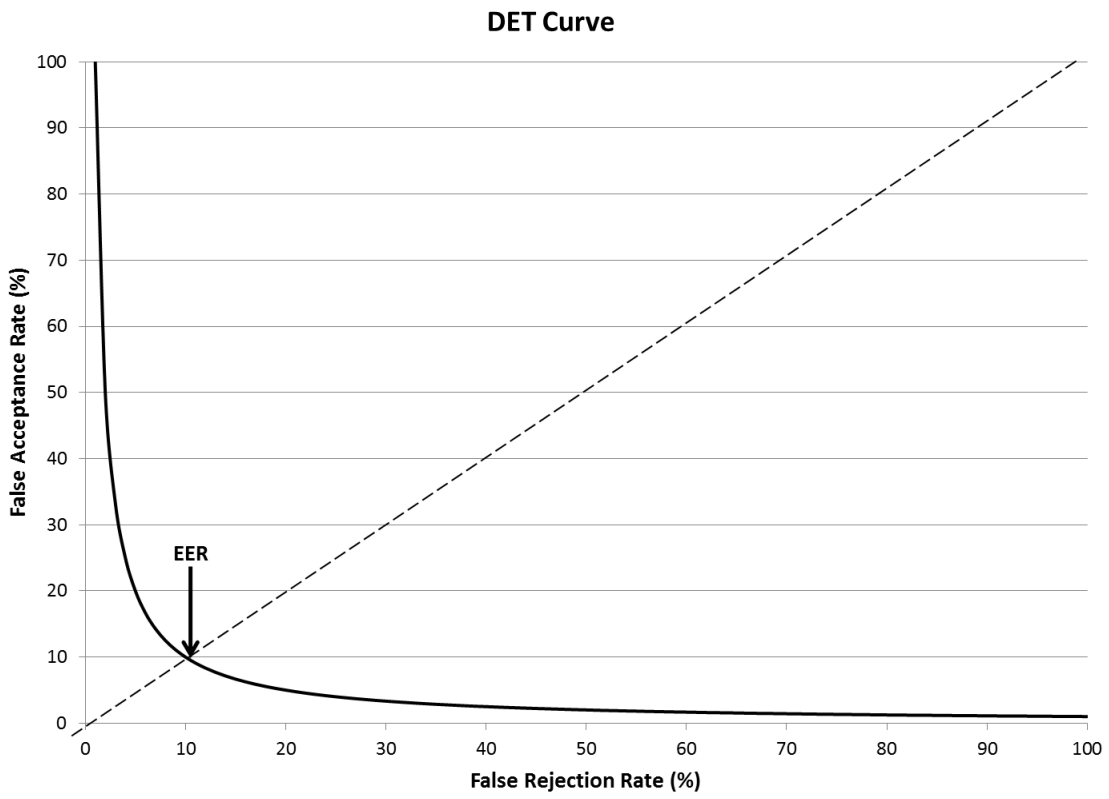

Figure 3.1: This figure demonstrates an example of a DET Curve with an EER around 10%.

The following subsections describe the design of the GRF biometric system implemented to achieve our research objectives and explain the limitations and assumptions imposed on the system.

3.1.1 Recognition Techniques

The experiment performed by our GRF biometric system facilitates novel research that addresses some of the previously identified GRF-recognition research gaps. In the area of GRF normalization two new GRF normalization techniques were designed for our research: Localized Least Squares Regression (LLSR) and Localized Least Squares

Regression with Dynamic Time Warping (LLSRDTW). These techniques are compared with the L^{∞} normalization and LTN techniques used in previous GRF recognition studies in addition to the popular L^1, L^2, and Z-Score normalizers. Unlike previous studies, the research in this thesis also compares normalized to non-normalized GRF recognition performance. Furthermore, it increases the GRF classification knowledge-base by extending the list of classifiers applied to GRF recognition to include the promising new Least Squares Probabilistic Classification (LSPC) classifier, a novel discriminative classification technique first proposed by Sugiyama in 2010 [42]. Additionally, as demonstrated in table 3.1, the experimental research presented in this thesis, for the first time, compares the GRF recognition performance of the wavelet packet feature extraction technique to the spectral and holistic techniques, the holistic feature extraction technique to the spectral, and the MLP to the LDA classifier.

As well as expanding upon previous GRF recognition techniques, the biometric system presented here also makes it possible to compare the impact of variations in shoe type against each technique (chapter 7). With such information it should be possible to identify the recognition techniques that best mitigate any of potentially negative effects that variation in shoe type might have on GRF recognition. Details regarding the optimization of this experiment are presented through the following three chapters, results are collected in the chapter 7, and an assessment of the acquired results is presented in chapter 8.

Techniques	Previous GRF Recognition Research
Feature Extractors	
Geometric	[31] [7] [32] [3] [4]
Holistic	[3] [39] [4] [30]
Spectral	[5] [32]
Wavelet Packet	[7]
Normalizers	
L^{∞}	[3] [4]
LTN	[7]
L^{1}, L^{2}, Z-Score, LLSR, LLSRDTW	(not previously studied)
Classifiers	
KNN	[5] [39] [31] [7] [32] [3]
MLP	[32]
SVM	[7] [3] [32] [4]
LDA	[7]
LSPC	(not previously studied)

Table 3.1: This table provides a comparison of GRF-recognition techniques examined across previous GRF-recognition studies with those examined in this thesis.

3.1.2 Experimental Biometric System

The design flow of the system, demonstrated in figure 3.2, made it possible for two high level experiments to be carried out, one on a development dataset and another on an evaluation dataset. For each of these datasets, the system was setup so that every possible configuration of feature extractor, normalizer, and classifier could be tested. The processing of the development dataset in figure 3.2 includes an additional optimization step during enrolment prior to training. This additional step involves using the development dataset to discover normalizer, feature extractor, and classifier parameters that correspond to optimal recognition results for the respective dataset; with the hope these parameters will also optimize for any unseen evaluation data. Once optimization is complete, enrolment in either dataset is accomplished by first passing the training samples through a normalizer (optionally) and feature extractor to acquire a feature set,

and then feeding the resulting sample feature sets to a classifier so that appropriate subject boundaries can be learned. The enrolment phase is considered complete once the system is fully trained. Later, during the challenge phase, the testing subset is transformed again using the chosen normalizer and feature extractor, then all possible combinations of correct and incorrect verification requests from the testing subset are run against the trained classifier.

The classifiers in this system are configured to return the probability that a verification request is correct; in classifiers that do not naturally return a posterior probability this probability is estimated using the un-scaled output values (described in chapter 6). The generated probability reflects the likelihood that a provided sample matches the given verification request; this makes it possible to set an acceptance threshold, with every returned verification probability greater-than-or-equal to the acceptance threshold accepted by the classifier, and those less-than the threshold rejected. Comparing the accepted and rejected verification requests with the expected results exposes the FAR and FRR, and, with a variable acceptance threshold, it is possible to tune the results in such a way that the difference between the FAR and FRR is minimized and the EER can be approximated. Finally, using a technique known as K-fold cross validation [43], the whole process is repeated against all other possible training and testing subset pairs with dimensions matching those of the initial pairs. The final experimental result returned by this biometric system contains the EER that was found by adjusting a single acceptance threshold to simultaneously minimize the difference between the FAR and FRR across each of the dataset's cross validation sample spaces.

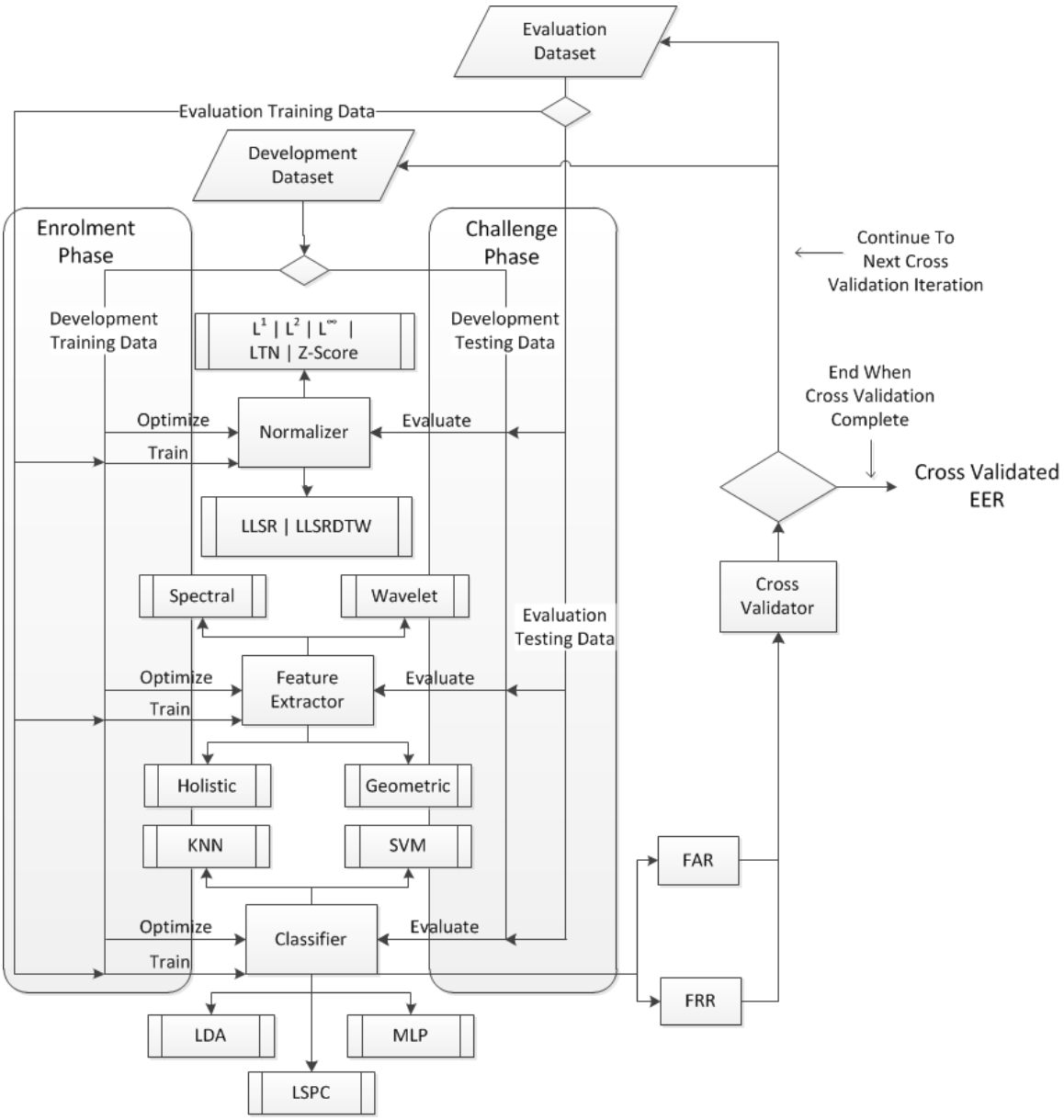

Figure 3.2: GRF-Recognition Biometric System Design. This diagram demonstrates the normalizers, feature extractors and classifiers used by the GRF Biometric System. The datasets are split into two subsets and processed in two phases with final classification results returned in verification mode. The evaluation set is only run against development set-optimized configurations to give results with reduced bias and explore alternative data formations. The process uses a cross validator for increased accuracy.

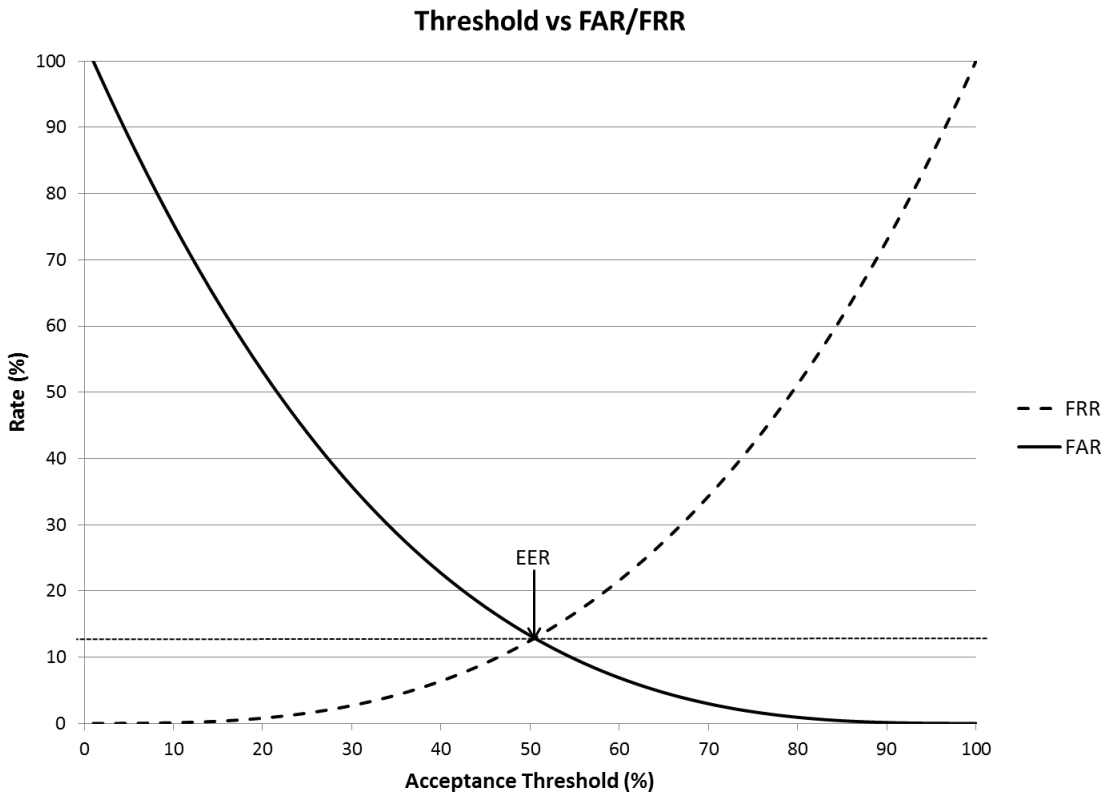

Figure 3.3: This diagram demonstrates how an EER can be obtained by adjusting the classifier acceptance threshold. In this case the EER is found when the acceptance threshold is set to about 50%.

The implementation of the experimental biometric system used in our research was accomplished using a Microsoft Visual Studio 2010 C# solution consisting of four projects. The solution includes one external library and incorporates code from a number of different sources, referenced in chapters 4 through 6. The experimental data used by the system, detailed in section 3.2, consists of a series of pre-collected GRF stepping force data files. The high level project structure is demonstrated in figure 3.4. The tester project is responsible for reading in data files, training the system, and evaluating configurations of the system using cross validation. The feature generator applies the techniques used to normalize the data prior to feature extraction (chapter 5), as well as

those used to actually extract the features (chapter 4). The normalization generator takes features extracted from the feature generator and formats them appropriately for use in a classifier. To complete the classification process, the formatted features extracted from the normalization generator, are passed into a chosen classifier contained within the classification generator project (chapter 6). This setup allows for various biometric system configurations to be compared on a single platform, reducing the potential for bias in experimental results.

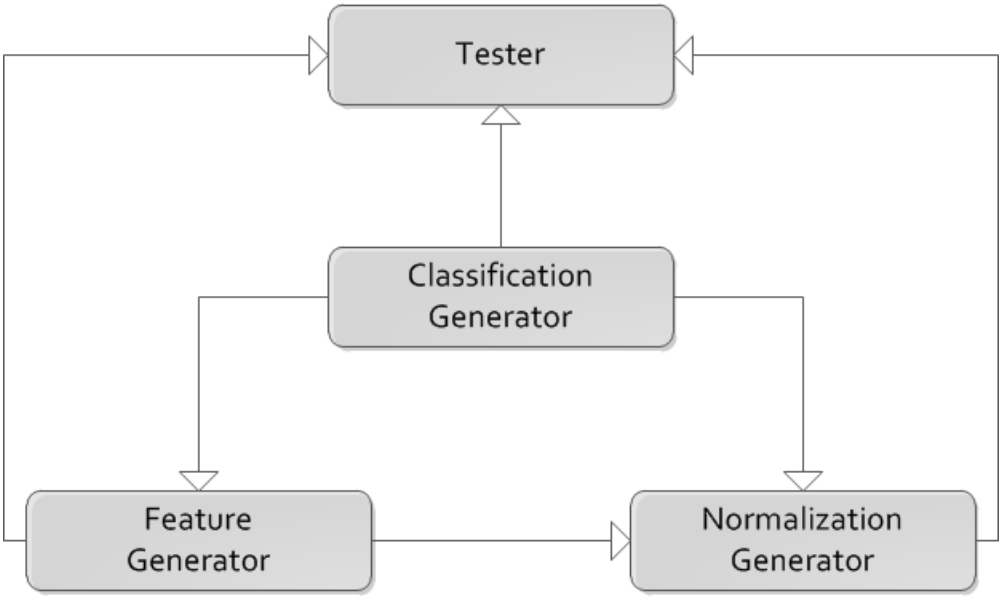

Figure 3.4: This figure demonstrates a high level project implementation for the GRF biometric system.

3.1.3 Experiment Scope

The analysis performed in this thesis is based on several assumptions. First, the experimental results presented in this thesis are applicable only under assumption that an

individual's footstep GRF remains consistent throughout its entire recognition use period. In practical applications this assumption may fail due to injuries, impairment, or significant differences in footwear. It has also been suggested that an individual's gait is likely to change as he or she ages [44]. Furthermore, this thesis defines a footstep as beginning from a heel-plant and progressing in a rolling motion to a toe push as shown in figure 3.5; other variations of footstep are not considered. Moreover, footsteps, for the purpose of this thesis, are assumed to come from walking persons rather than running persons. When a person is walking there will always be at least one foot on the ground, however, during a running motion both feet will be off the ground for a period of time. This is important because the type of motion has a dramatic effect on the shape of the resulting GRF signature, as demonstrated in figure 3.6.

The research in this thesis also has been limited to a dataset containing only 10 different individuals. During the biometric system enrolment phase, information from each individual is used to train the system. Additionally, the data collected for this research was obtained from cooperative participants landing clean footsteps on a force plate. Therefore, it can be assumed that any results obtained came under nearly ideal conditions, and, in practice, with the potential for non-cooperative and non-enrolled participants weaker performance should be expected.

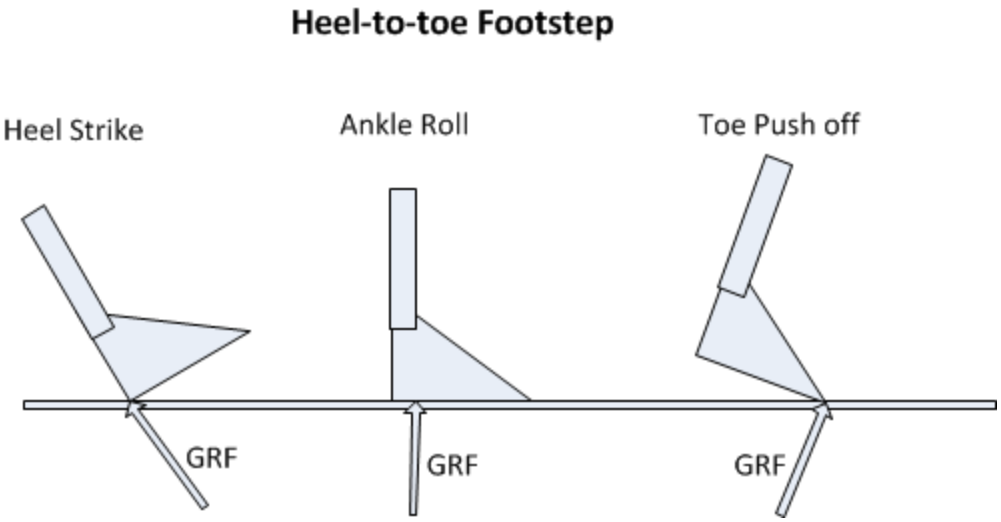

Figure 3.5: This diagram demonstrates the three primary stances during a heel-to-toe footstep.

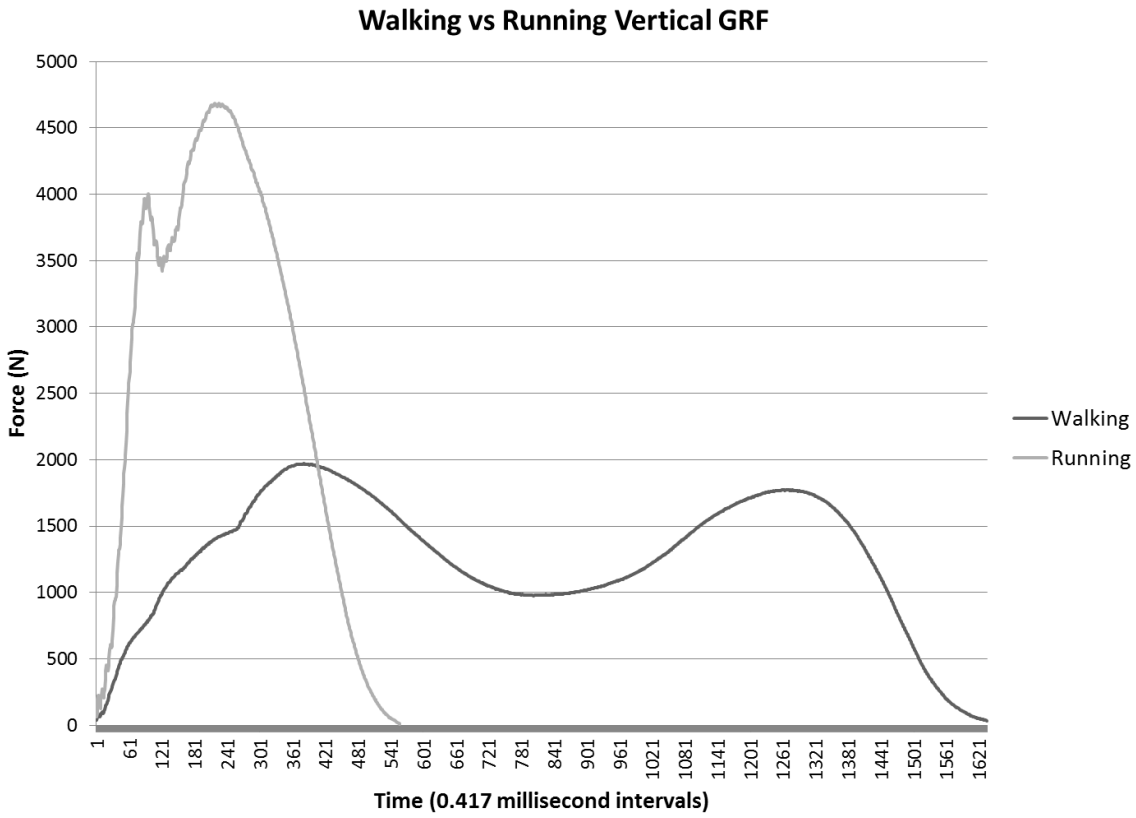

Figure 3.6: This figure provides a comparison of the walking to the running vertical GRF component in the same individual.

3.2 Experimental Data

The dataset used in this thesis is represented as two smaller sets, the development and evaluation datasets. The set used in the following three chapters to train and optimize GRF recognition techniques is referred to as the development dataset. In chapter 7 the performance of the various GRF recognition techniques and impact of shoe type variation is measured on previously unseen data in the evaluation dataset. Both datasets contain footstep samples from the same 10 subjects, but vary in the shoe type used during sample collection. The development dataset consists of 10 footstep samples per subject, taken with all subjects wearing Asics runners, whereas the evaluation dataset consists of two sets with 10 footstep samples per subject, one with all subjects wearing Orin runners and the other with all subjects wearing Verona runners.

The experimental data specifications, demonstrated in table 3.2, were built around data samples provided by the University of Calgary Faculty of Kinesiology. The provided data samples came from 10 different male athletes ranging in age from 21 to 30 years old. Each individual was asked to achieve a walking speed of approximately 1.5 m/s and make a clean step over a Kistler Force Plate [38] apparatus. The Kistler Force Plate is rectangular and contains sensors on each of its four corners. Using different combinations of these sensors the apparatus returns 8 different output signals: two representing the GRF anterior-posterior component, two representing the GRF medial-lateral component, and four representing the GRF vertical component. The research presented in this thesis makes use of all 8 output signals covering all three of the GRF components, which, in

contrast to the previous studies shown in table 3.3, allows the GRF dynamics to be examined in greater detail.

Experimental Data

Apparatus	Kistler Force Plate	
Sampling Rate	2400Hz	
Output Signals		
GRF Vertical Component (FZ)	GRF Anterior-Posterior Component (FY)	GRF Medial-Lateral Component (FX)
4	2	2
Data Size	10 people	
Total Samples	300	
Walking Speed	1.5 m/s	
Duration	20 s	

	Samples Per Person	**Shoe Type**
Development Dataset		
	10	Asics Runner
Evaluation Dataset		
	10	Orin Runner
	10	Verona Runner

Table 3.2: This table demonstrates the experiment data specifications. The 8 output signals used in this experiment are shown according to the GRF component they belong to; there are 4 output signals belonging to the vertical component, 2 in the anterior-posterior component, and 2 in the medial-lateral component.

Research Group	**GRF Components Analyzed**	**Data Output Signals**
Addlesee et al. [30]	FZ	1
Orr and Abowd [31]	FX,FY,FZ	1
Cattin [5]	FZ	1
Suutala and Röning [32]	FX,FY,FZ	1
Moustakidis et al. [7]	FX,FY,FZ	3
Rodríguez et al. [3,4]	FX,FY,FZ	1
Mostayed et al. [39]	FZ	1

Table 3.3: This table provides a comparison of data samples components used by previous research groups.

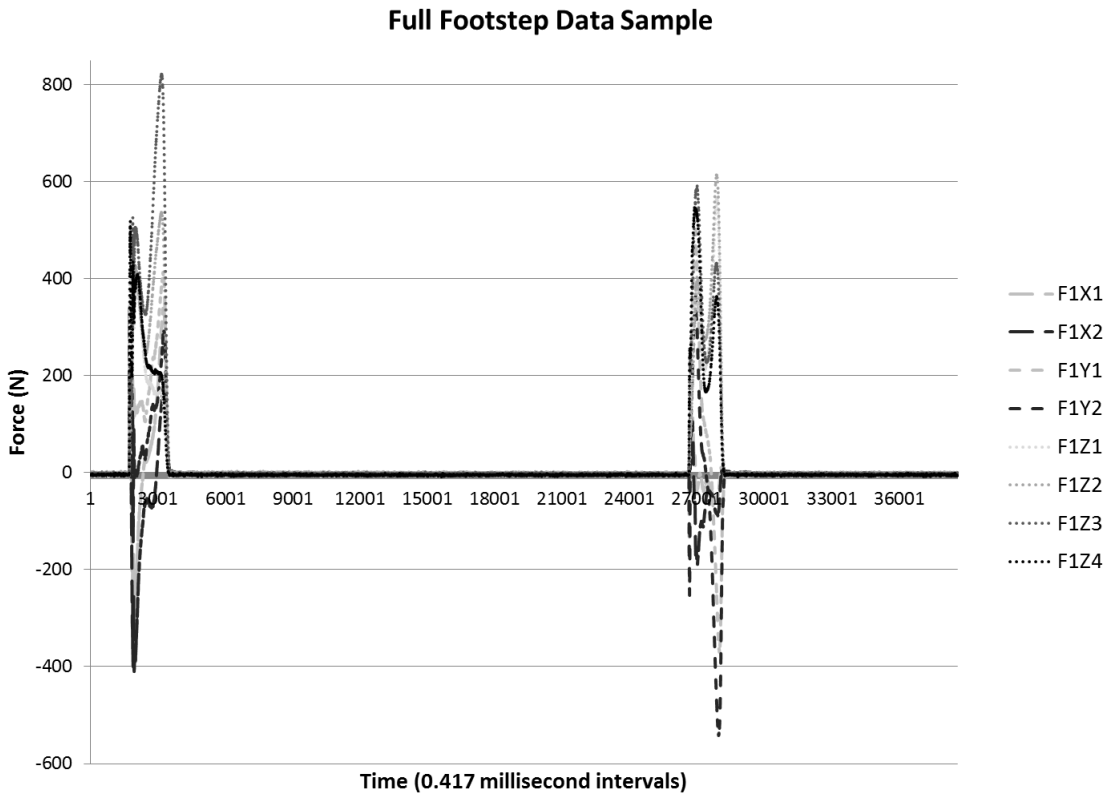

Figure 3.7: This figure shows the data representation returned by the Kistler Force Plate, including a synchronization footstep followed by the footstep used for the research in this thesis.

The raw output of a single sample collected from the force plate is demonstrated in figure 3.7. The sample is approximately 20 seconds long and contains two distinct force spikes. Of these two force spikes, only the second spike contained useful footstep GRF information, while the first was used for an unrelated synchronization purpose. As shown in the diagram, data from the 8 different output signals was collected with a 0.417 millisecond sampling frequency and varied between positive and negative values. The output signals labelled F1X1 through F1Z4 represent the breakdown of the three GRF components. The readings between the two force spikes were not entirely smooth, but rather, appeared to reflect small frequent vibrations from the floor.

To effectively analyze the various proposed GRF-recognition techniques, it was first necessary to isolate the desired footstep signature from the remaining extraneous data. The process of footstep extraction presented several challenges: first the footstep extractor needed to know when to start extracting the footstep, next the extractor needed to know when to stop extracting the footstep, and finally the extractor needed to extract only the second foot step rather than the first synchronization force spike. To satisfy these conditions for footstep extraction a simple process was devised.

To develop the step extractor process it was first necessary to examine the sample data in closer detail. After examining the start and end points of several footsteps in the sample sets it was clear that all samples shared similar characteristics. It also became apparent that the force on signals labelled F1Z1 through F1Z4 remained positive and amplified for the entire duration of a footstep. Using this finding, it was determined that the signals F1Z1 through F1Z4 would be easiest to use to establish a threshold for starting and stopping the extraction process.

The start and end points of a sample footstep are demonstrated in figure 3.8 below. While the start of the footstep shows a sudden well defined force spike in the Z-labelled signals, the end of the footstep is marked by a slow decline in amplitude, resulting in it being more susceptible to error from environmental noise. Experimental trials showed that starting the extraction of the footstep when the force in any of the Z-labelled signals exceeded a threshold of 15N was sufficient for capturing footstep data while also generally ignoring any noise between steps. However, ending the extraction when any of

the Z-labelled signals fell below the 15N threshold was more problematic, as the signals would often bounce back above the 15N threshold shortly after, resulting in the extractor falsely believing that a new step had occurred. To account for this residual trailing noise it was determined through experimentation, that, once all Z-labelled signals fell below the 15N threshold and all failed rise above the threshold after 15 sampling intervals, the extraction process could be safely terminated.

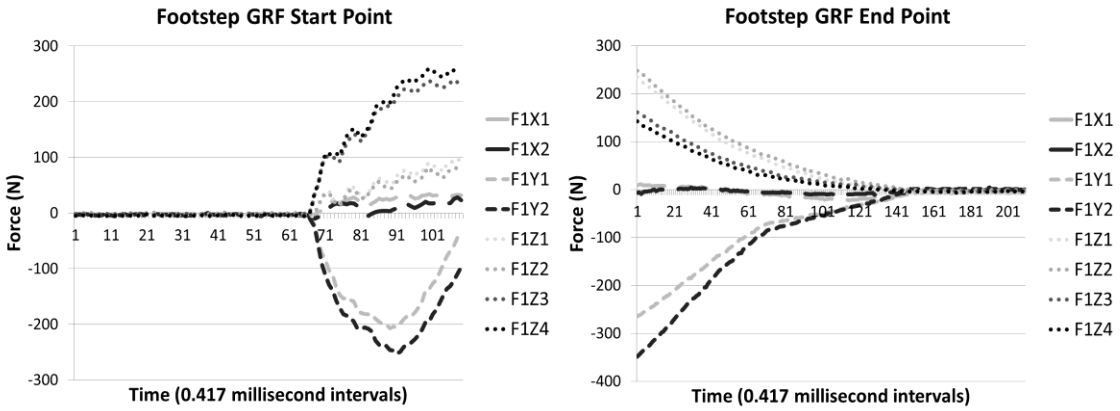

Figure 3.8: This figure provides an example of the beginning and end of the footstep GRF

The footstep extraction process applied in this experiment is described below. During experimentation one sample was found to have random brief spikes of force, so an extra step, step 3, was added to discard data when these occurred.

1) Loop through every row in the provided data sample. For each row if any FZ output signal is greater than 15N then start recording.
2) While recording, if all FZ signals have remained less than 15N for 15 iterations then stop recording.

3) Discard any recording that ends up less than 100 rows long (these are likely due to error and actual steps will be around 1600 rows long).
4) Two recordings should remain at the end of the process: the first representing the synchronization force spike and the second representing the actual footstep force spike. Discard the synchronization recording and we are left with only the data from the desired footstep sample.

After applying the above process to the experimental dataset, the extracted footsteps were labelled according to their owner's age, with a numeric qualifier appended to the end of the label to distinguish different persons belonging to the same age group (i.e. m22_2). The extraction process was able to successfully extract footsteps in all but one of the 300 available data samples. The unsuccessful sample, the 10th Orin sample for the m22_1 subject, was determined to contain an incomplete footstep.

Experiment Footstep Data Parameters

	m21_1	m22_1	m22_2	m22_3	m25_1	m25_2	m25_3	m27_1	m28_1	m30_1
Asics	10	10	10	10	10	10	10	10	10	10
Verona	10	10	10	10	10	10	10	10	10	10
Orin	10	9	10	10	10	10	10	10	10	10

Training Set Size: 5 samples per person

Table 3.4: This table shows the available footstep data and predetermined training subset size for this experiment. The numbers under each person refer to the number of sample footsteps acquired for that person for the corresponding shoe type.

To perform recognition analysis, the dataset must be broken up into training and testing subsets. While, to avoid potential bias, the size of the training set should be the same for

all tests run. In this experiment it was decided that a training set of 5 samples per person would be sufficient for assessing the impact of various recognition techniques.

3.3 Summary

This chapter introduced the concept of the biometric system and presented an experimental design for a system with the objective of performing footstep-based GRF person recognition. The system was designed to incorporate varying combinations of feature extractors, normalizers, and classifiers. Among these components were the novel LLSR and LLSRDTW normalizers developed in our research (discussed in chapter 5), and the LSPC classifier, a classifier never before used for the purpose of footstep GRF recognition (discussed in chapter 6). This chapter also presented the methods to be used for measuring recognition performance in the remainder of this thesis, selecting the verification EER as the metric of choice, and, to clarify assumptions and limitations with regards to the problem domain an experiment problem scope was covered.

Having presented the experimental design, the chapter continued on to describe the dataset for which our chosen biometric system was configured to work. Following best practices, the dataset was split into a development dataset, to be used in the optimization and analysis of each biometric component, and a mutually exclusive evaluation dataset to be used in obtaining results less influenced by training bias. The three chapters that follow build on the high level methodology introduced here by demonstrating the theoretical backgrounds, optimization techniques, and implementations applied for each

of the feature extraction, normalization, and classification components used in the formation of our biometric system.

Chapter 4

Feature Extraction

Discovering the footstep GRF characteristics that have a unique range in values for any given person would make it possible, in the absence of spoofing, to perform recognition with perfect accuracy using even the simplest of classifiers. When dealing with data samples containing thousands of recorded values (also referred to as dimensions), leaving the identification of these characteristics to classifiers alone can be computationally expensive and potentially lead to classifier overfitting, which occurs when undesirable characteristics, such as noise, are misinterpreted as being significant during training. One way to address these issues is to preprocess the data using a technique known as feature extraction.

Feature extraction aims to represent the characteristics that best distinguish the original dataset in a reduced dimensional space. The objectives of feature extraction are closely aligned with those of data compression. However, the data compression requirement that enough information be retained to be able to reconstruct the original dataset to some chosen degree of accuracy does not apply to feature extraction. For the purpose of footstep GRF recognition the goal is to extract a feature set that minimizes the degree of feature space overlap between any two people. This chapter presents the principles behind our four chosen GRF feature extraction techniques and explains how each was configured to accommodate our dataset. Additionally, the work demonstrated in this

chapter examines a number of methods used to optimize each extraction technique for better recognition performance, building upon work done in previous research.

4.1 Geometric

A number of previous studies [31, 7, 32, 3, 4] have identified spatial and statistical footstep GRF characteristics that can be extracted to form a feature space. These heuristically derived characteristics are referred to as geometric features. Spatial features include specific data measurements such as the position of local maxima or displacement between two points of interest, while statistical features reflect the properties of the dataset taken as a whole and include measurements like the mean GRF value. The sample shown in figure 4.1 demonstrates the force values of the 8 output signals over the course of a footstep and has identified the points of interest, the local minima and maxima determined to be consistent enough to be used as features.

Using the points of interest identified in figure 4.1 together with the geometric features proposed in [4], our research was able to identify 538 potential geometric features across the 8 output signals. Of these features, 506 were spatial characteristics and 32 were statistical characteristics. The spatial features included time and force values for all 22 extrema points identified in figure 4.1, the 231 displacements in time between every pair of extrema points, and the 231 displacements in force between every pair of extrema points. The statistical features were restricted to force measurements only and included the mean values, areas under the curve, standard deviations, and norms for each of the 8 output signals. A breakdown of the features is shown in table 4.1.

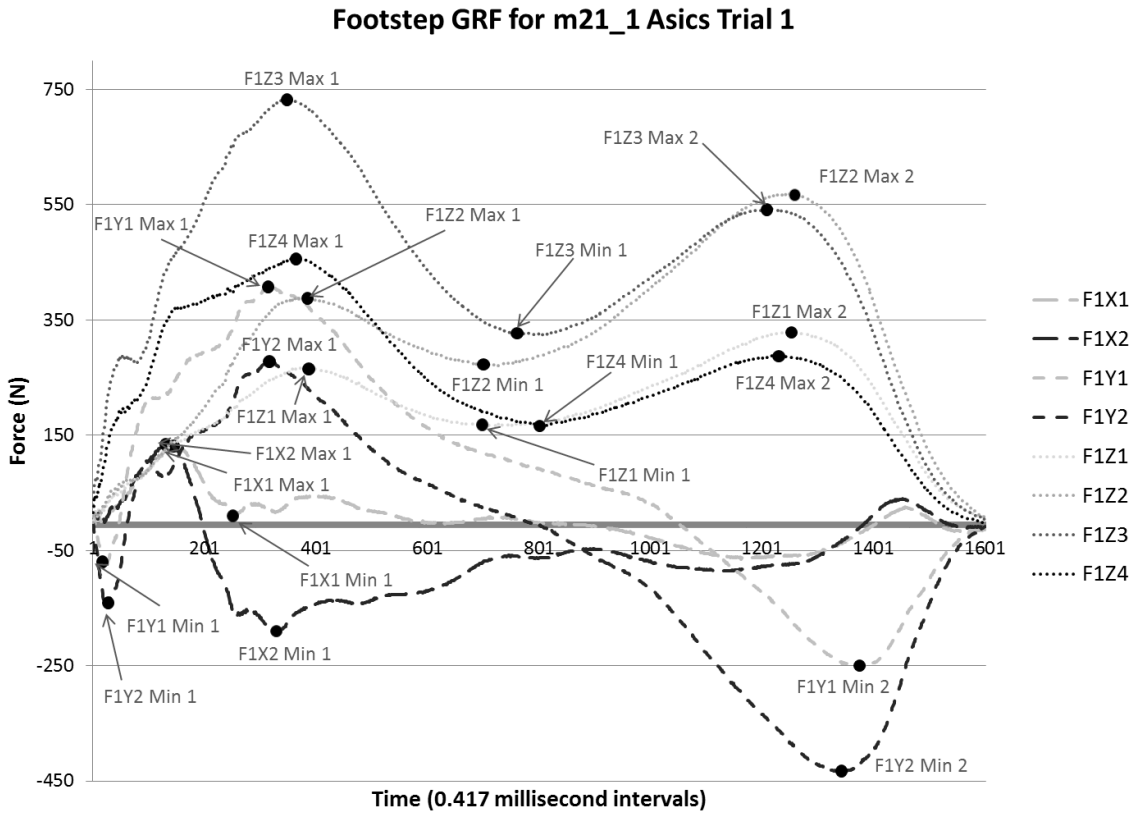

Figure 4.1: This diagram demonstrates the set of heuristically derived points of interest across our 8 GRF signals. Each point is identified with a dot and given a descriptive label.

Spatial Point Features

Output Signal	Measure	Features
F1X1	Force	Max1, Min1
F1X1	Time	Max1, Min1
F1X2	Force	Max1, Min1
F1X2	Time	Max1, Min1
F1Y1	Force	Min1, Max1, Min2
F1Y1	Time	Min1, Max1, Min2
F1Y2	Force	Min1, Max1, Min2
F1Y2	Time	Min1, Max1, Min2
F1Z1	Force	Max1, Min1, Max2
F1Z1	Time	Max1, Min1, Max2
F1Z2	Force	Max1, Min1, Max2
F1Z2	Time	Max1, Min1, Max2
F1Z3	Force	Max1, Min1, Max2
F1Z3	Time	Max1, Min1, Max2
F1Z4	Force	Max1, Min1, Max2
F1Z4	Time	Max1, Min1, Max2

Spatial Displacement Features

Output Signal(s)	Features
F1X1	$D_{2}(F_1X_1\,Force),\ D_{2}(F_1X_1\,Time)$
F1X1 - F1X2	$D_{2}(F_1X_1\,Force \cup F_1X_2\,Force) - (F_1X_1\,Force) - (F_1X_2\,Force),\ D_{2}(F_1X_1\,Time \cup F_1X_2\,Time) - (F_1X_1\,Time) - (F_1X_2\,Time)$
F1X1 - F1Y1	$D_{2}(F_1X_1\,Force \cup F_1Y_1\,Force) - (F_1X_1\,Force) - (F_1Y_1\,Force),\ D_{2}(F_1X_1\,Time \cup F_1Y_1\,Time) - (F_1X_1\,Time) - (F_1Y_1\,Time)$
F1X1 - F1Y2	$D_{2}(F_1X_1\,Force \cup F_1Y_2\,Force) - (F_1X_1\,Force) - (F_1Y_2\,Force),\ D_{2}(F_1X_1\,Time \cup F_1Y_2\,Time) - (F_1X_1\,Time) - (F_1Y_2\,Time)$
F1X1 - F1Z1	$D_{2}(F_1X_1\,Force \cup F_1Z_1\,Force) - (F_1X_1\,Force) - (F_1Z_1\,Force),\ D_{2}(F_1X_1\,Time \cup F_1Z_1\,Time) - (F_1X_1\,Time) - (F_1Z_1\,Time)$
F1X1 - F1Z2	$D_{2}(F_1X_1\,Force \cup F_1Z_2\,Force) - (F_1X_1\,Force) - (F_1Z_2\,Force),\ D_{2}(F_1X_1\,Time \cup F_1Z_2\,Time) - (F_1X_1\,Time) - (F_1Z_2\,Time)$
F1X1 - F1Z3	$D_{2}(F_1X_1\,Force \cup F_1Z_3\,Force) - (F_1X_1\,Force) - (F_1Z_3\,Force),\ D_{2}(F_1X_1\,Time \cup F_1Z_3\,Time) - (F_1X_1\,Time) - (F_1Z_3\,Time)$
F1X1 - F1Z4	$D_{2}(F_1X_1\,Force \cup F_1Z_4\,Force) - (F_1X_1\,Force) - (F_1Z_4\,Force),\ D_{2}(F_1X_1\,Time \cup F_1Z_4\,Time) - (F_1X_1\,Time) - (F_1Z_4\,Time)$
F1X2	$D_{2}(F_1X_2\,Force),\ D_{2}(F_1X_2\,Time)$
F1X2 - F1Y1	$D_{2}(F_1X_2\,Force \cup F_1Y_1\,Force) - (F_1X_2\,Force) - (F_1Y_1\,Force),\ D_{2}(F_1X_2\,Time \cup F_1Y_1\,Time) - (F_1X_2\,Time) - (F_1Y_1\,Time)$
F1X2 - F1Y2	$D_{2}(F_1X_2\,Force \cup F_1Y_2\,Force) - (F_1X_2\,Force) - (F_1Y_2\,Force),\ D_{2}(F_1X_2\,Time \cup F_1Y_2\,Time) - (F_1X_2\,Time) - (F_1Y_2\,Time)$
F1X2 - F1Z1	$D_{2}(F_1X_2\,Force \cup F_1Z_1\,Force) - (F_1X_2\,Force) - (F_1Z_1\,Force),\ D_{2}(F_1X_2\,Time \cup F_1Z_1\,Time) - (F_1X_2\,Time) - (F_1Z_1\,Time)$
F1X2 - F1Z2	$D_{2}(F_1X_2\,Force \cup F_1Z_2\,Force) - (F_1X_2\,Force) - (F_1Z_2\,Force),\ D_{2}(F_1X_2\,Time \cup F_1Z_2\,Time) - (F_1X_2\,Time) - (F_1Z_2\,Time)$
F1X2 - F1Z3	$D_{2}(F_1X_2\,Force \cup F_1Z_3\,Force) - (F_1X_2\,Force) - (F_1Z_3\,Force),\ D_{2}(F_1X_2\,Time \cup F_1Z_3\,Time) - (F_1X_2\,Time) - (F_1Z_3\,Time)$
F1X2 - F1Z4	$D_{2}(F_1X_2\,Force \cup F_1Z_4\,Force) - (F_1X_2\,Force) - (F_1Z_4\,Force),\ D_{2}(F_1X_2\,Time \cup F_1Z_4\,Time) - (F_1X_2\,Time) - (F_1Z_4\,Time)$
F1Y1	$D_{2}(F_1Y_1\,Force),\ D_{2}(F_1Y_1\,Time)$
F1Y1 - F1Y2	$D_{2}(F_1Y_1\,Force \cup F_1Y_2\,Force) - (F_1Y_1\,Force) - (F_1Y_2\,Force),\ D_{2}(F_1Y_1\,Time \cup F_1Y_2\,Time) - (F_1Y_1\,Time) - (F_1Y_2\,Time)$
F1Y1 - F1Z1	$D_{2}(F_1Y_1\,Force \cup F_1Z_1\,Force) - (F_1Y_1\,Force) - (F_1Z_1\,Force),\ D_{2}(F_1Y_1\,Time \cup F_1Z_1\,Time) - (F_1Y_1\,Time) - (F_1Z_1\,Time)$
F1Y1 - F1Z2	$D_{2}(F_1Y_1\,Force \cup F_1Z_2\,Force) - (F_1Y_1\,Force) - (F_1Z_2\,Force),\ D_{2}(F_1Y_1\,Time \cup F_1Z_2\,Time) - (F_1Y_1\,Time) - (F_1Z_2\,Time)$
F1Y1 - F1Z3	$D_{2}(F_1Y_1\,Force \cup F_1Z_3\,Force) - (F_1Y_1\,Force) - (F_1Z_3\,Force),\ D_{2}(F_1Y_1\,Time \cup F_1Z_3\,Time) - (F_1Y_1\,Time) - (F_1Z_3\,Time)$
F1Y1 - F1Z4	$D_{2}(F_1Y_1\,Force \cup F_1Z_4\,Force) - (F_1Y_1\,Force) - (F_1Z_4\,Force),\ D_{2}(F_1Y_1\,Time \cup F_1Z_4\,Time) - (F_1Y_1\,Time) - (F_1Z_4\,Time)$
F1Y2	$D_{2}(F_1Y_2\,Force),\ D_{2}(F_1Y_2\,Time)$
F1Y2 - F1Z1	$D_{2}(F_1Y_2\,Force \cup F_1Z_1\,Force) - (F_1Y_2\,Force) - (F_1Z_1\,Force),\ D_{2}(F_1Y_2\,Time \cup F_1Z_1\,Time) - (F_1Y_2\,Time) - (F_1Z_1\,Time)$
F1Y2 - F1Z2	$D_{2}(F_1Y_2\,Force \cup F_1Z_2\,Force) - (F_1Y_2\,Force) - (F_1Z_2\,Force),\ D_{2}(F_1Y_2\,Time \cup F_1Z_2\,Time) - (F_1Y_2\,Time) - (F_1Z_2\,Time)$
F1Y2 - F1Z3	$D_{2}(F_1Y_2\,Force \cup F_1Z_3\,Force) - (F_1Y_2\,Force) - (F_1Z_3\,Force),\ D_{2}(F_1Y_2\,Time \cup F_1Z_3\,Time) - (F_1Y_2\,Time) - (F_1Z_3\,Time)$
F1Y2 - F1Z4	$D_{2}(F_1Y_2\,Force \cup F_1Z_4\,Force) - (F_1Y_2\,Force) - (F_1Z_4\,Force),\ D_{2}(F_1Y_2\,Time \cup F_1Z_4\,Time) - (F_1Y_2\,Time) - (F_1Z_4\,Time)$
F1Z1	$D_{2}(F_1Z_1\,Force),\ D_{2}(F_1Z_1\,Time)$
F1Z1 - F1Z2	$D_{2}(F_1Z_1\,Force \cup F_1Z_2\,Force) - (F_1Z_1\,Force) - (F_1Z_2\,Force),\ D_{2}(F_1Z_1\,Time \cup F_1Z_2\,Time) - (F_1Z_1\,Time) - (F_1Z_2\,Time)$
F1Z1 - F1Z3	$D_{2}(F_1Z_1\,Force \cup F_1Z_3\,Force) - (F_1Z_1\,Force) - (F_1Z_3\,Force),\ D_{2}(F_1Z_1\,Time \cup F_1Z_3\,Time) - (F_1Z_1\,Time) - (F_1Z_3\,Time)$
F1Z1 - F1Z4	$D_{2}(F_1Z_1\,Force \cup F_1Z_4\,Force) - (F_1Z_1\,Force) - (F_1Z_4\,Force),\ D_{2}(F_1Z_1\,Time \cup F_1Z_4\,Time) - (F_1Z_1\,Time) - (F_1Z_4\,Time)$
F1Z2	$D_{2}(F_1Z_2\,Force),\ D_{2}(F_1Z_2\,Time)$
F1Z2 - F1Z3	$D_{2}(F_1Z_2\,Force \cup F_1Z_3\,Force) - (F_1Z_2\,Force) - (F_1Z_3\,Force),\ D_{2}(F_1Z_2\,Time \cup F_1Z_3\,Time) - (F_1Z_2\,Time) - (F_1Z_3\,Time)$
F1Z2 - F1Z4	$D_{2}(F_1Z_2\,Force \cup F_1Z_4\,Force) - (F_1Z_2\,Force) - (F_1Z_4\,Force),\ D_{2}(F_1Z_2\,Time \cup F_1Z_4\,Time) - (F_1Z_2\,Time) - (F_1Z_4\,Time)$
F1Z3	$D_{2}(F_1Z_3\,Force),\ D_{2}(F_1Z_3\,Time)$
F1Z3 - F1Z4	$D_{2}(F_1Z_3\,Force \cup F_1Z_4\,Force) - (F_1Z_3\,Force) - (F_1Z_4\,Force),\ D_{2}(F_1Z_3\,Time \cup F_1Z_4\,Time) - (F_1Z_3\,Time) - (F_1Z_4\,Time)$
F1Z4	$D_{2}(F_1Z_4\,Force),\ D_{2}(F_1Z_4\,Time)$

Statistical Features

Output Signal	Measure	Features
F1X1	Force	Area, Mean, Standard Deviation, Norm
F1X2	Force	Area, Mean, Standard Deviation, Norm
F1Y1	Force	Area, Mean, Standard Deviation, Norm
F1Y2	Force	Area, Mean, Standard Deviation, Norm
F1Z1	Force	Area, Mean, Standard Deviation, Norm
F1Z2	Force	Area, Mean, Standard Deviation, Norm
F1Z3	Force	Area, Mean, Standard Deviation, Norm
F1Z4	Force	Area, Mean, Standard Deviation, Norm

Table 4.1: The three sub-tables above demonstrate the spatial and statistical features examined in our research. Features measured in force refer to measurements in Newtons, while time features refer to measurements in Seconds. Displacement features were too numerous to display in this table and therefore were represented as sets using the binomial coefficient notation to express set membership. For example, set of distances between every 2-feature combination from the four F1X1 Force point features would be represented as $\boldsymbol{D_{\binom{F1X1\ Force}{2}}}$.

Having identified the geometric features of interest, the next challenge was to construct an algorithm capable of extracting these features from the GRF curves. The greatest difficultly in this regard involved locating desirable local extrema points on a signal containing a substantial level of noise. To acquire these points, two procedures were developed: a local maxima locator and a local minima locator. Initially the procedures were setup to accept some initialization point from the GRF data series and iterate forward from that point until the values began to either decrease, when looking for a local maxima, or increase, when looking for a local minima. The point at which the values started to increase or decrease was returned as the local minima or maxima, respectively. This procedure proved problematic as incorrect extrema points were often returned due to noise in the data and footstep imperfections. To address this problem the procedure was redesigned to accept two additional parameters: one to make the algorithm less sensitive

to undesirable small peaks or troughs in the data, and another to “smooth” the data to more accurately determine the exact location of each extrema point.

The pseudo-code for the local maxima locator procedure is demonstrated in figure 4.2. The minima procedure is almost identical but the inequality on line 5 is reversed. This procedure tracks the maximum value as the force values increase along the ridge, but when the values start to decrease, rather than immediately returning the last maxima found, the procedure continues iterating until the sensitivity threshold is reached. If the sensitivity threshold is reached then the maximum value recorded before the threshold counter began would be the largest value found so far and therefore would be returned as the local maxima. While the sensitivity threshold returned the greatest value as the local maxima, the value returned often turned out to be a single-record spike in force due to noise, rather than the actual visual top of the force ridge. To reduce the impact of noise a previous gait-based study by Derawi et al. [28] imposed a weighted moving average on the data. For our research, a simple moving average was determined to be sufficient for noise reduction. When data smoothing is used, rather than looking for the single value maxima, the procedure looks for an averaged multi-value maxima and large spikes in data are smoothed into a more level plane; this increases the likelihood that any maxima found by the procedure will indeed be the actual top of the ridge.

Input: Initial GRF Index (X_{init}, Y_{init}), Integer threshold, Integer smoothing
N = Sample Size
Output: The Next Local Maxima (X_{max}, Y_{max})

```
1   smooth(Y_point) ← Average(Y_point– smoothing/2 … Y_point + smoothing/2
2   Y_max ← Y_init
3   cnt ← 0
4   for Y_current = Y_smoothing … Y_N
5       if smooth(Y_current) > Y_max
6           Y_max ← Y_current
7           X_max ← X_current
8           cnt ← 0
9       else
10          cnt ← cnt + 1
11          if cnt > threshold
12              return (X_max, Y_max)
13          end
14      end
15  end
16  return (X_max, Y_max)
```

Figure 4.2: This figure demonstrates the pseudo-code for the algorithm we used for the local maxima locator.

To locate the full set of point features on each output signal the local minima and maxima locator procedures were chained together with the point located by the one locator forming the initialization index for the next in the chain. For example, in the Z-labelled output signals, the process would start at the first data record in the footstep then move forward through the records to locate the first local maximum. Next, the first local maximum would be assigned as the initialization index for the local minima locator, which, in turn, would locate the next local minimum. Finally, the local minimum returned by the local minima locator would be passed back into the local maxima locator as the initialization index, and the process would terminate with the locations of the first local maximum, the first local minimum, and the second local maximum all identified. When applied to the development dataset, this process was able to correctly locate the point

features in all but one output signal in a single footstep sample, using sensitivity threshold parameter values in the range of 200-300 intervals for the Z/Y-labelled signals and 30 intervals for the X-labelled signal together with a smoothing parameter value of 5.

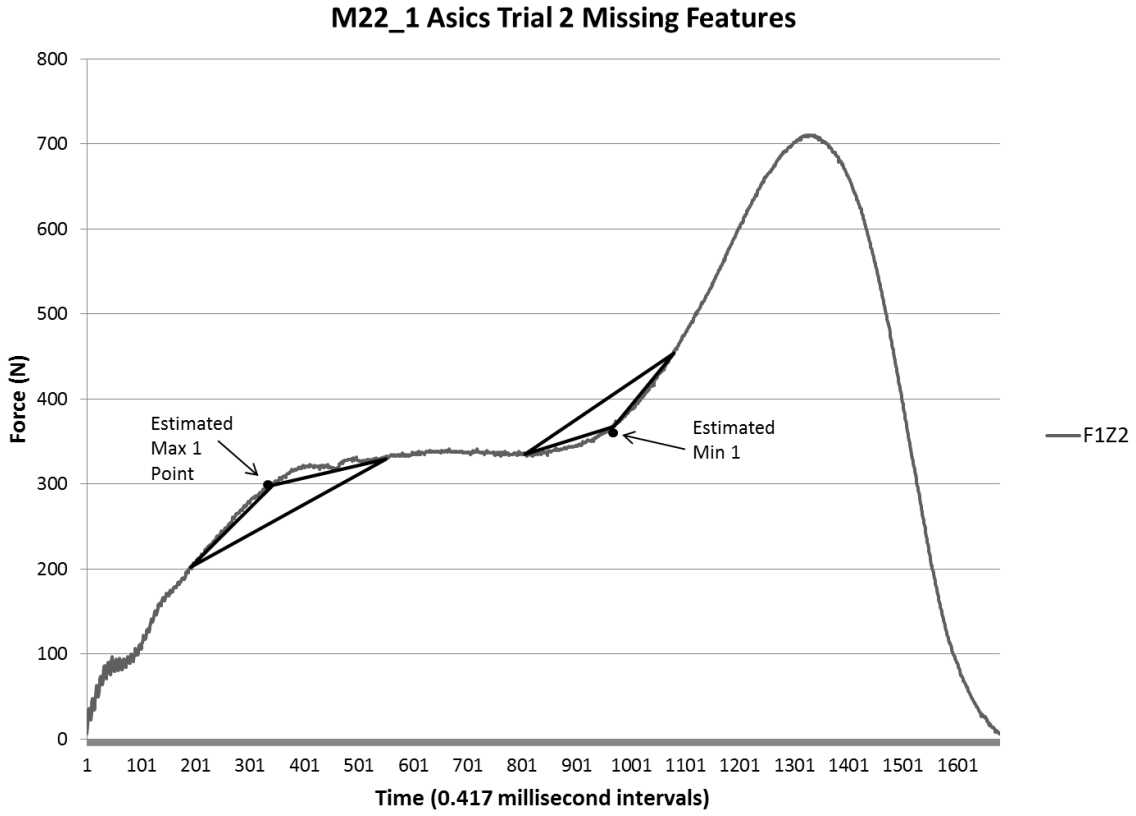

Figure 4.3: This figure demonstrates the triangle approximation for missing point features. The points that maximized the area of the inscribed triangles were approximated as the expected position of the missing local extrema features.

A visual inspection of the sample with the missing point features revealed that its F1Z2 output signal did not contain a maximum point where the expected first maximum would typically occur. Consequently, the minimum value point was also undefined. However, the graph, shown in figure 4.3, still demonstrated a well-defined convex curvature in the region where the first local maximum would be expected and concave curvature where

the following local minimum would be expected. Thus, rather than leaving these features out, we decided to estimate the points of maximum convex and concave curvature, then use these points to approximate the expected positions of the local maxima and minima in the case that the initial locator process failed. Our estimation approach was based on a study by Castellanos et al. [45] that aimed to estimate the corner of the L-curve. In their work, they estimated that, given three points on a curve forming a triangle, the point of maximum curvature would be the position at which the middle of these three points had a minimum angle value. In our implementation we used a slightly different approach and searched for the three points, evenly spread across a 200 record-wide region of the graph, which maximized the area of the triangle formed by connecting each point. Again, the middle of these three points was taken as the approximate point of maximum curvature, as demonstrated in figure 4.3. The points that resulted from applying this technique to missing features-sample yielded a local maximum of (355, 307) and local minimum of (1012, 394). The estimated maximum point appeared relatively close to its average position of (368, 397) for the subject M22_1, while the estimated minimum point displayed significant error when compared to its average position of (736, 322). Because this missing feature showed up in only 1 of 300 data samples, the error was considered acceptable.

Having captured the full set of spatial point features for each sample, capturing remaining displacement and statistical features was relatively simple. The displacements were captured by finding the difference in time and force between the pairs formed by every combination of points, as listed in table 4.1. The mean, Euclidean norm and standard

deviation were acquired by applying their respective statistical formulas to the set of all force recordings in the sample space, while the area under the curve was approximated by applying the trapezoidal rule to the sample space (equation 4.1).

$$\int_a^b f(x)dx \approx \frac{1}{2}\sum_{k=1}^{N}(x_{k+1} - x_k)(f(x_{k+1}) + f(x_k))$$

Equation 4.1: This equation demonstrates the trapezoidal approximation of an integral using the Trapezoidal Rule.

When the values for individual features in our 538 geometric feature space were compared, it became apparent that some features were much better discriminators than others. Furthermore, some features contained so much variability that using them for classification would likely lead to overfitting and decrease recognition performance. In [4], Rodríguez et al. suggested an optimization technique to remove undesirable features from the geometric feature set. To accomplish this they used an exhaustive search process. The process started by identifying the stand-alone feature that produced the smallest EER during classification, then searched for the feature that, when combined with the initially discovered feature, produced the smallest feature pair EER. This process continued until the full feature set was sorted such that, when grouped with all the features ahead of it, each feature in the sorted set produced better performance than any of the features behind it. Using this process Rodríguez's team found that, with only the 17 best of their initial 42 geometric features, they were able to reduce their EER by 22%.

Building upon the work of [4], our research incorporated k-fold cross validation into the original process. In our case, rather than sorting features based on the best EER for a single training/testing subsets pair, the EER was calculated by taking the EER produced across all 10 possible training/testing subsets in our development dataset. Our approach also differed in that we performed the optimization using a weighted KNN classifier (see chapter 6), rather than the SVM classifier used in [4]. In this optimization, as well as all our other feature extraction optimizations and analysis, the K parameter was set to an arbitrary value of 5. The final optimization process functioned as follows:

1) Let **O** be an empty set that will contain all geometric features ordered from most to least discriminative. Let **G** be the full geometric feature space.
2) Take a feature from **G** that is not currently in **O** and add it to **O**.
3) For each training/testing subset in the development dataset calculate the EER.
4) Find the EER across every training/testing subset, and then remove the feature that was added in Step 2.
5) Repeat Step 2-4 for every feature not in **O**.
6) Add the feature with the best averaged EER (Step 4) to **O** then repeat Step 2 until **O** contains every feature in **G**.
7) Take the first *N* features that best reduce the EER and feature space dimensionality. These features will form the optimized geometric dataset.

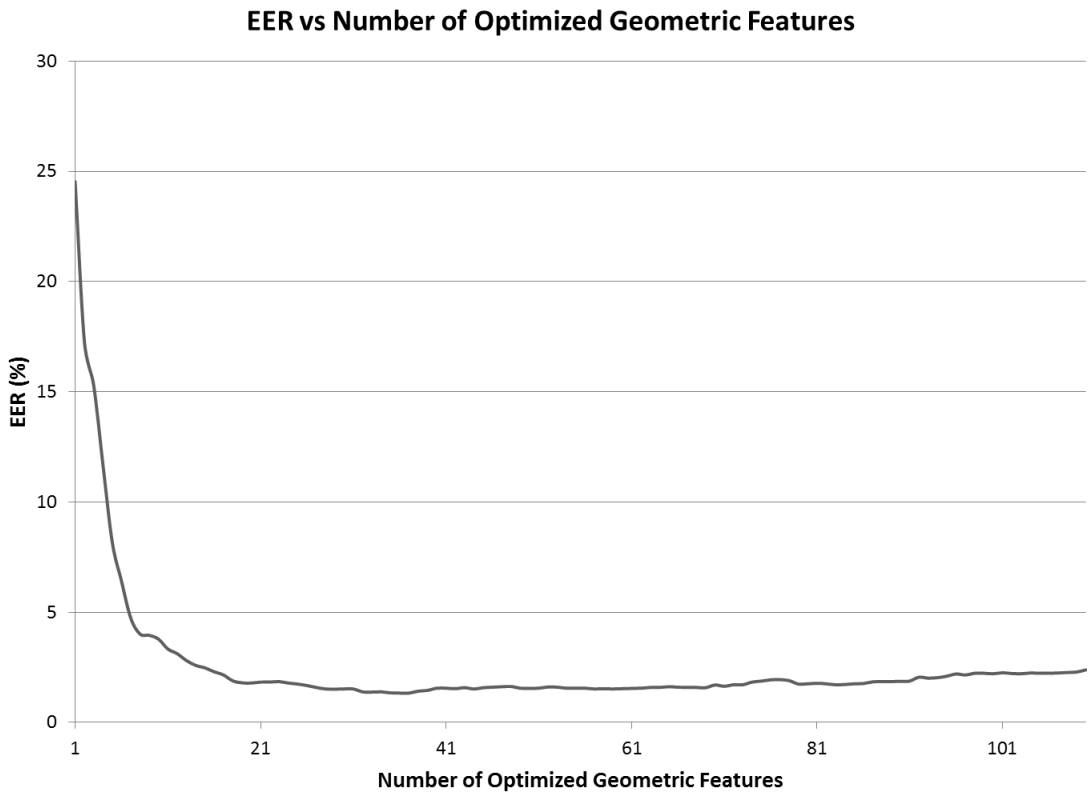

Figure 4.4: This graph compares the number of optimized features to the EER they produced. The count of features starts with the best performing single feature, then the best performing two features and continues with the features added resulting in smaller increases in performance as the size of the feature set grows.

The optimization process, when run on the original 538-feature geometric feature space using the development dataset, resulted in a significant improvement in recognition performance. The change in EER for the 110 initial optimization iterations is shown in figure 4.4. The diagram demonstrates a sharp drop in the EER up to the point when the optimal feature set contains 21 features, after which the EER flattens and even begins to increase as the optimal feature set grows in size. This suggests classifier overfitting begins to hinder recognition in large geometric feature spaces. Consequently, for the purpose of this thesis, the first 36 optimal features were chosen to form the optimal

geometric feature space. This optimal feature space contained 6 spatial point features, 26 displacement features, and 4 statistical features, roughly corresponding to the frequency of each feature type in the original space. Furthermore, the measurement units for the optimal feature space consisted of about as many time features as there were force features. While, compared with the original geometric feature space, the optimal feature space decreased the cross validated development dataset EER from 8.48% to 1.33333%, equivalent to an 84% increase in performance. Additionally, the 36-feature optimized geometric feature space represented a 93% decrease in dimensionality over the full geometric feature space and a 99.7% decrease over the roughly 12800 record full footstep GRF data space.

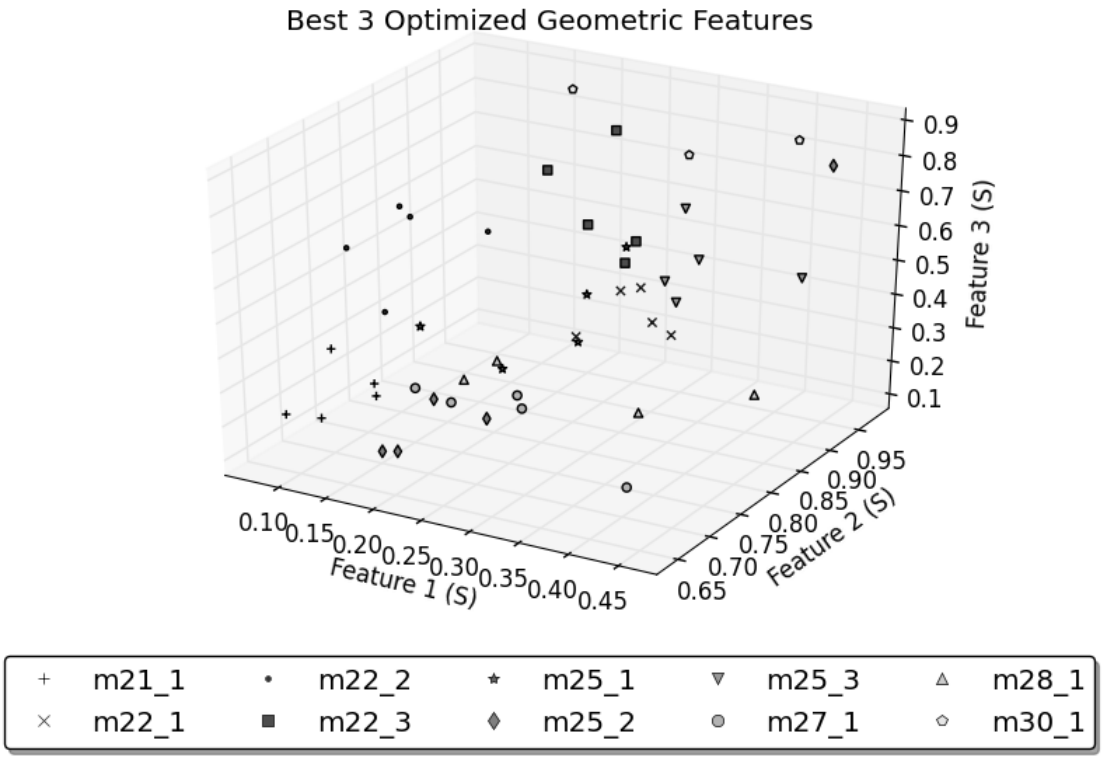

Figure 4.5: This diagram presents a visualization of the 3 best ranked optimized geometric features taken from footsteps belonging to the ten test subjects and projected into a 3d

frame. Five footsteps per person are shown in this diagram and the footsteps belonging to each subject are distinguished by variations in the marker symbols used. For better visualization the range for each feature has been standardized as [0,1].

Optimal Geometric Features

Feature	Unit	Feature	Unit
$D_{F1Z4MAX1\_F1X2MIN1}$	TIME	$D_{F1Y1MAX1\_F1Y2MIN2}$	TIME
F1X2 MIN1	TIME	F1X2 MAX1	FORCE
$D_{F1Z2MAX2\_F1Y1MIN2}$	TIME	F1Z3 MAX1	FORCE
$D_{F1Y1MIN2\_F1X1MAX1}$	FORCE	F1X1 MEAN	FORCE
F1Y2 NORM	FORCE	F1X1 MAX1	TIME
$D_{F1X2MAX1\_F1X2MIN1}$	FORCE	$D_{F1Z3MAX1\_F1X2MAX1}$	TIME
$D_{F1Y2MIN1\_F1X2MAX1}$	TIME	$D_{F1Y2MIN2\_F1X1MAX1}$	FORCE
$D_{F1Z3MAX2\_F1Y1MIN2}$	TIME	$D_{F1Y1MIN1\_F1Y2MIN1}$	FORCE
$D_{F1Z1MAX1\_F1Z1MAX2}$	FORCE	$D_{F1Z3MIN1\_F1X2MIN1}$	TIME
$D_{F1Z1MAX2\_F1Y1MIN2}$	TIME	$D_{F1Z1MAX2\_F1Y2MIN2}$	TIME
$D_{F1Z2MAX1\_F1Z2MAX2}$	TIME	$D_{F1Z2MAX1\_F1Z4MAX2}$	TIME
F1Y1 MIN1	FORCE	F1X1 AREA	FORCE
$D_{F1Z2MAX1\_F1Z2MAX2}$	FORCE		
F1Y1 MIN2	FORCE		
$D_{F1Y2MAX1\_F1X2MAX1}$	FORCE		
F1Y2 STDEV	FORCE		
$D_{F1Y1MIN2\_F1X2MAX1}$	TIME		
$D_{F1Y1MIN2\_F1Y2MIN2}$	FORCE		
$D_{F1Y2MAX1\_F1X2MAX1}$	TIME		
$D_{F1Z1MIN1\_F1X2MIN1}$	TIME		
$D_{F1Y1MIN1\_F1X2MAX1}$	TIME		
$D_{F1X1MAX1\_F1X2MAX1}$	TIME		
$D_{F1Z1MIN1\_F1Z1MAX2}$	FORCE		
$D_{F1Z1MIN1\_F1X1MAX1}$	TIME		

Table 4.2: This table demonstrates the best 36 features in the feature set remaining after the geometric dataset was optimized. Point features are identified by the output signal name and extrema point type, while displacement features are identified with a 'D' and a subscript containing the two point features that formed the displacement. Each feature is categorized as either a measure of force (Newtons) or time (Seconds).

Feature Space Comparison

Feature Space	Cross Validated EER (%)	Dimensions
Geometric	8.47777	538
Optimal Geometric	1.33333	36

Table 4.3: This table compares the performance of geometric feature spaces on the development dataset.

4.2 Holistic

Feature extraction techniques defined by heuristics, including the geometric technique described in the previous section, can be powerful data discriminators, but rely on expert knowledge to identify the most valuable characteristics in a dataset. When datasets are complex or have not been thoroughly studied, heuristic-based techniques can suffer. The alternative to heuristic-based feature extraction approaches involves using machine learning techniques to discover important data characteristics. These techniques do the work of identifying the most statistically significant characteristics in a dataset, according to some pre-defined learning model. For the purpose of this thesis, the non-heuristic techniques have been categorized into those that only perform feature extraction after a transformation of the data domain has occurred and those that only work with the data in its original domain. We refer to the latter category of feature extractor as holistic techniques following the usage of the term in [3, 4], and in our case the raw data being processed is represented in the time domain.

As noted in the previous chapter, several existing footstep GRF recognition studies [3, 39, 4, 30] have implemented holistic feature extraction solutions. In its simplest form, the holistic approach involves no feature extraction at all; instead entire data samples are passed as-is for classification and the determination of important characteristics is left to the classifier. Unfortunately, with each of our data samples consisting of approximately 12800 records, passing the raw data to a classifier would be computationally undesirable and would likely lead to massive classifier overfitting and very poor recognition performance. To address this problem we have based our holistic feature extraction on

the dimensionality reducing holistic technique proposed by Rodríguez et al. in [3]. In their work they proposed a way to reduce the dimensionality of the original dataset using Principal Component Analysis (PCA). Using PCA, Rodríguez's team discovered 80 features, from 4200 in their original dataset, accounted for 96% of the dataset's original information (variance). When running PCA on a dataset with multiple persons and without a high level of variation between samples of a single individual, because variance represents variability in the dataset, we would expect to see the features with the greatest variance across the entire dataset also represent the features with the greatest disparity between individuals; the discovery of such distinguishing features would be important to achieve strong recognition performance.

PCA involves transforming the original feature space, in our case the approximately 12800 record holistic sample space, into a new feature space where the new features, or Principal Components (PCs), are uncorrelated and ordered according to the amount of variance they represent [46]. The greatest challenge when implementing PCA comes from the need to generate a transformation that is able to represent data in the PC feature space. Once this transformation has been generated, any given sample can be projected into the new PC feature space and dimensionality can be reduced by extracting only the small group of PCs that account for the majority of the original feature space's variance. A five step process to generate the needed PCA transform is described by Lindsey Smith in [47]. The process begins by calculating the mean for each feature in the training subset, and then proceeds to subtract the calculated means from the respective features in each training sample; the end result is a dataset whose mean is zero. Next, the covariance

matrix for the zero-mean dataset is calculated according to equation 4.2. Once we have the covariance matrix, we need to calculate its eigenvectors and eigenvalues; these can be found by first solving for the eigenvalues (equation 4.4) then substituting each eigenvalue back into equation 4.3 to solve for the eigenvectors. When all eigenvalues and eigenvectors have been calculated, we will find the size of the eigenvalues corresponds to their respective degrees of variance in the original dataset; with this knowledge we can extract a small set of eigenvectors that accounts for the majority of the variance in the original dataset to form our PCs. Finally, when we want to project data samples from our original feature space into the new smaller PC feature space we can do so using this small set of eigenvectors and applying equation 4.5.

$$A^{n\times n} = (c_{i,j}, c_{i,j} = cov(Dim_i, Dim_j))$$

Equation 4.2: In this equation *A* represents the *n*×*n* covariance matrix, where Dim_x represents the x^{th} dimension.

$$Ax = \lambda x$$

Equation 4.3: To find the eigenvectors we must find all values of *x* that, when multiplied with our covariance matrix *A*, result in some scalar multiple *λ* of *x*. The vector solutions *x* form the eigenvectors, while the respective scalar multiples *λ* form the eigenvalues. If the *n*×*n* matrix has eigenvectors there will be *n* of them [47].

$$\det(A - \lambda I) = 0$$

Equation 4.4: With *A* being the *n*×*n* covariance matrix, *I* the *n*×*n* identity matrix, and *λ* the *nx1* eigenvalue matrix, if the covariance matrix has eigenvectors this equation will be applicable. This allows us to generate a characteristic polynomial equation to solve for the eigenvalues, which can then be substituted back into equation 4.3 to solve for the eigenvectors.

$$Y = W^T \times X^T$$

Equation 4.5: In this equation, *X* is a *1×n* input vector to be projected into PC-reduced feature space of size *m*. Each feature in *X* is assumed to have had its respective training subset-mean value subtracted from it. *W* is the *n×m* PC-reduced eigenvector matrix, which, when transposed contains the eigenvectors in the rows. Multiplying W^T with X^T we get *Y*, the *m×1* projection of *X* into the reduced feature space.

The implementation of the PCA feature extraction technique in our biometric system was accomplished through integration with the Accord.NET PCA library developed by César Souza [48]. This library featured several improvements over the process described in the previous paragraph. Rather than calculating the eigenvalues and eigenvectors directly, a task that can be computationally intensive, Souza acquired both sets using the computationally efficient Singular Value Decomposition (SVD) algorithm [49]. Souza's library also gave the option of using a correlation matrix instead of the covariance matrix when generating the PCs; an important option because the use of the correlation matrix can generate better performance when features have broad differences in their variances.

Before we could run PCA on our development dataset, we first needed to standardize the size of each data sample. Variations in stepping speed meant sample length and therefore the number of features per sample did not match up across the dataset. This was problematic for PCA, which expects a standard number of features to be present to both generate and carry out the feature space transform. To address this problem we followed the approach taken in [4], where an arbitrary number of records, large enough to represent full footsteps, were captured for each output signal. For our dataset, we determined that 2000 records per output signal, for a total of 16000 records per sample,

were sufficient to obtain all information of value in any given footstep. To ensure each sample contained the same number of records, extra zero-valued records were appended to signals with less than 2000 records. We refer to this approach as the point-based holistic approach.

Having established the process to perform PCA, when we wanted to perform classification using the dimensionality-reduced holistic feature extraction technique we did so through the following steps:

1) Perform PCA on the training data subset (i.e. 5 samples per person for all enrolled persons), and pick the best PCs to form the dimensionality-reducing feature space transform.
2) Project the training data subset into the new PC feature space.
3) Train the chosen classifier using the transformed training data samples.
4) Project a data sample from the testing subset into the new PC feature space.
5) Perform classification on the transformed testing data sample using the classifier from step 3.

The results from running the point-based holistic feature extraction technique on our KNN classifier are shown in figure 4.6. To better assess the accuracy of the point-based holistic approach, cross validation was performed with a unique PCA transform generated for every training/testing subset; the results in figure 4.6 reflect the average performance achieved across the 10 iterations of cross validation needed to cover our

entire development dataset. The best point-based holistic performance was found to occur when using the covariance matrix during PCA, and the optimal point-based holistic feature set contained the first 15 PCs accounting for approximately 98.7% of the dataset variance with an EER of 3.42%.

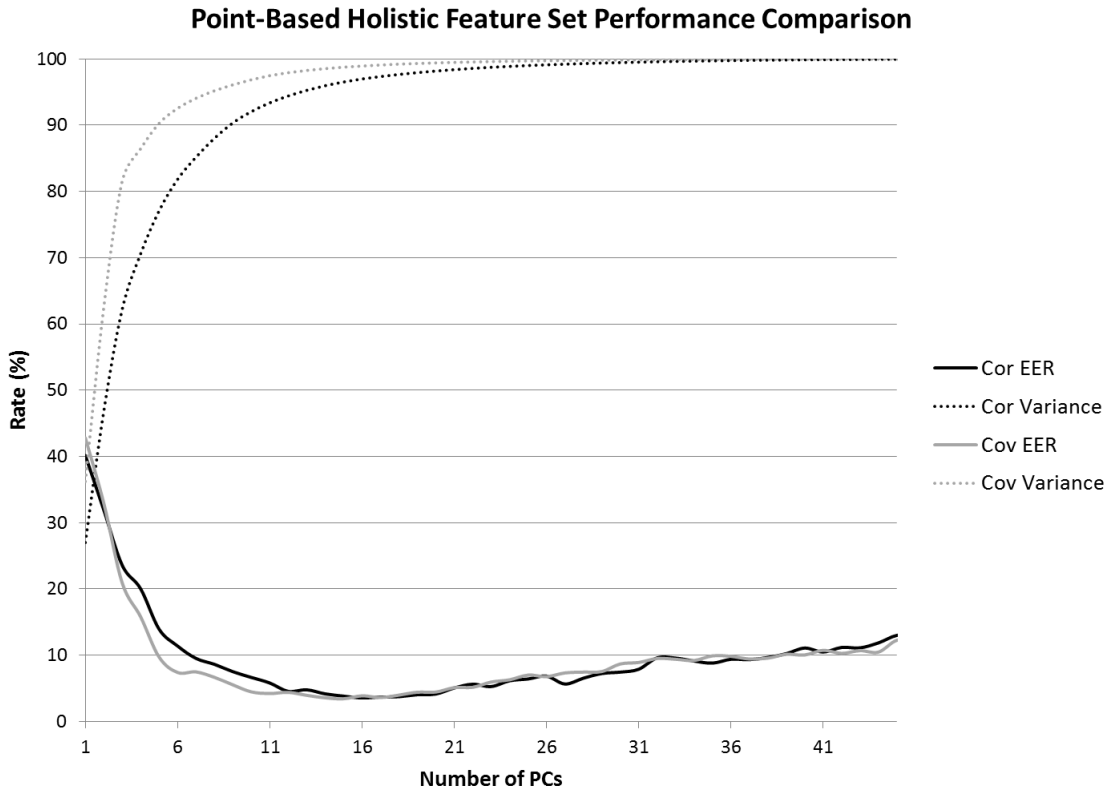

Figure 4.6: This figure demonstrates the performance of the point-based holistic approach using both covariance (cov) and correlation (cor) matrices during PCA. The PC set size is shown as a function of the EER it produces and the approximate dataset variance it represents.

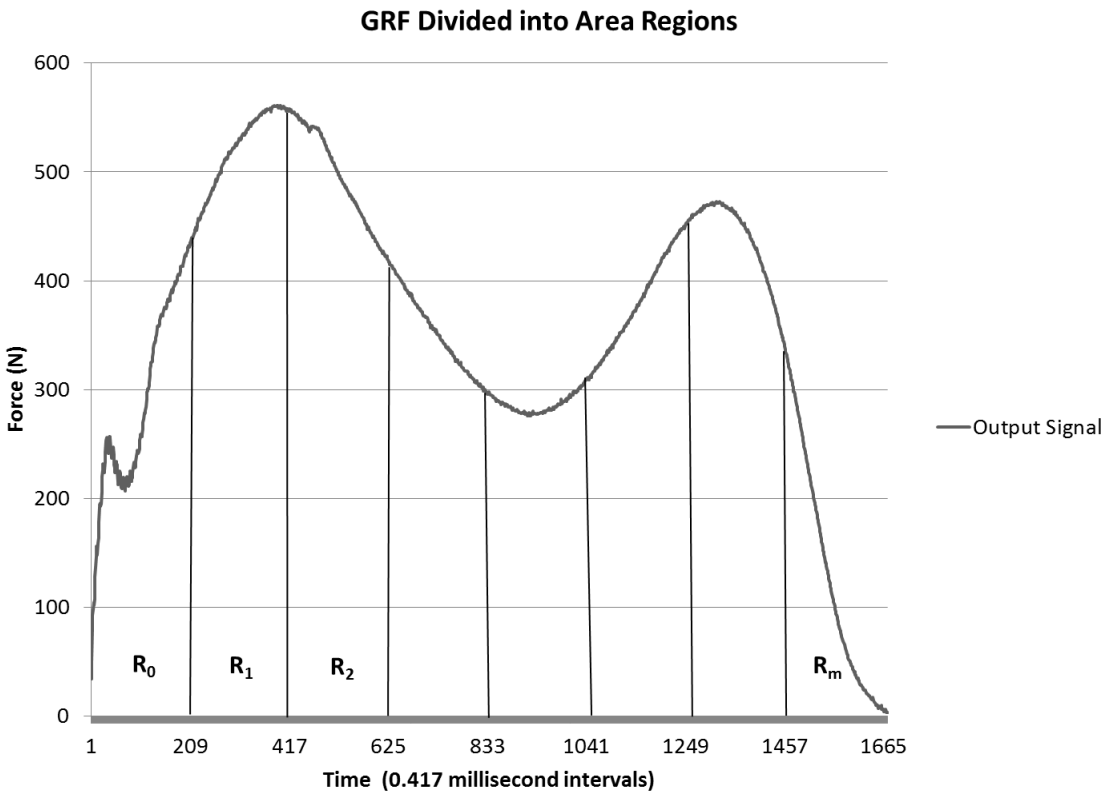

Figure 4.7: This figure demonstrates an example of the GRF data from one of our output signals divided into 8 regions. Using our area-based approach, the areas for each region would be calculated and stored as a new sample space.

In addition to the point-based holistic approach, during our research we developed a new approach for standardizing the size of footstep GRF sample space passed in for PCA. This new approach, which we refer to as the area-based holistic approach, involved dividing each sample into a standard number of temporally equal proportional regions and forming a new dataset with the set of regional areas. The graph in figure 4.7 shows how this would work on a single output signal with the dataset divided into 8 regional areas; in practice we would want far more regions for better resolution when performing PCA. Equations 4.6 through 4.9 demonstrate the process used to generate the new area-based sample space. To achieve a standard sample size for all available samples it is

important that the number of area regions is smaller than the number of points in the smallest expected sample; in our dataset no samples were less than 1400 records in length.

$$\boldsymbol{D} = ((t_0, f_0), (t_1, f_1), \dots (t_{N-1}, f_{N-1}))$$

Equation 4.6: Define the original data sample, of size *N*, as a series of records where t_i represents the time the sample occurred at and f_i represents the force value at t_i.

$$S(x) = \frac{N}{M} \times x$$

Equation 4.7: With *N* being the size of the origin sample space (*D*) and *M* being the size of the new area-based sample space, the function *S* returns the start of a given region *x*.

$$R_i = \frac{1}{2}\left(\lceil S(i) \rceil - S(i)\right)\left(t_{\lceil S(i) \rceil} - t_{\lceil S(i) \rceil - 1}\right)\left(f_{\lceil S(i) \rceil} + f_{\lceil S(i) \rceil - 1}\right)$$
$$+ \frac{1}{2}\sum_{j=\lceil S(i) \rceil}^{\lfloor S(i+1) \rfloor - 1}\left(t_{j+1} - t_j\right)\left(f_{j+1} + f_j\right)$$
$$+ \frac{1}{2}\left(\lfloor S(i+1) \rfloor - S(i+1)\right)\left(t_{\lfloor S(i+1) \rfloor + 1} - t_{\lfloor S(i+1) \rfloor}\right)\left(f_{\lfloor S(i+1) \rfloor + 1} + f_{\lfloor S(i+1) \rfloor}\right)$$

Equation 4.8: The area, R_i, for a given region *i*, is calculated using the sum of a set of trapezoidal approximations. Because area regions may start and/or end in the space between the records of the original sample, the equation was divided into three parts: the first part finds the approximate area for any partial region preceding the first original dataset record *(t, f)* in R_i, the second part calculates the approximate area for the set of original dataset records falling within R_i, and final part approximates the area for any partial region following the last original dataset record in R_i.

$$\boldsymbol{R} = (R_0, R_1, \dots R_{M-1}), M < N$$

Equation 4.9: To get the new area-based sample space we simply calculate R_i for *i = 1:M*.

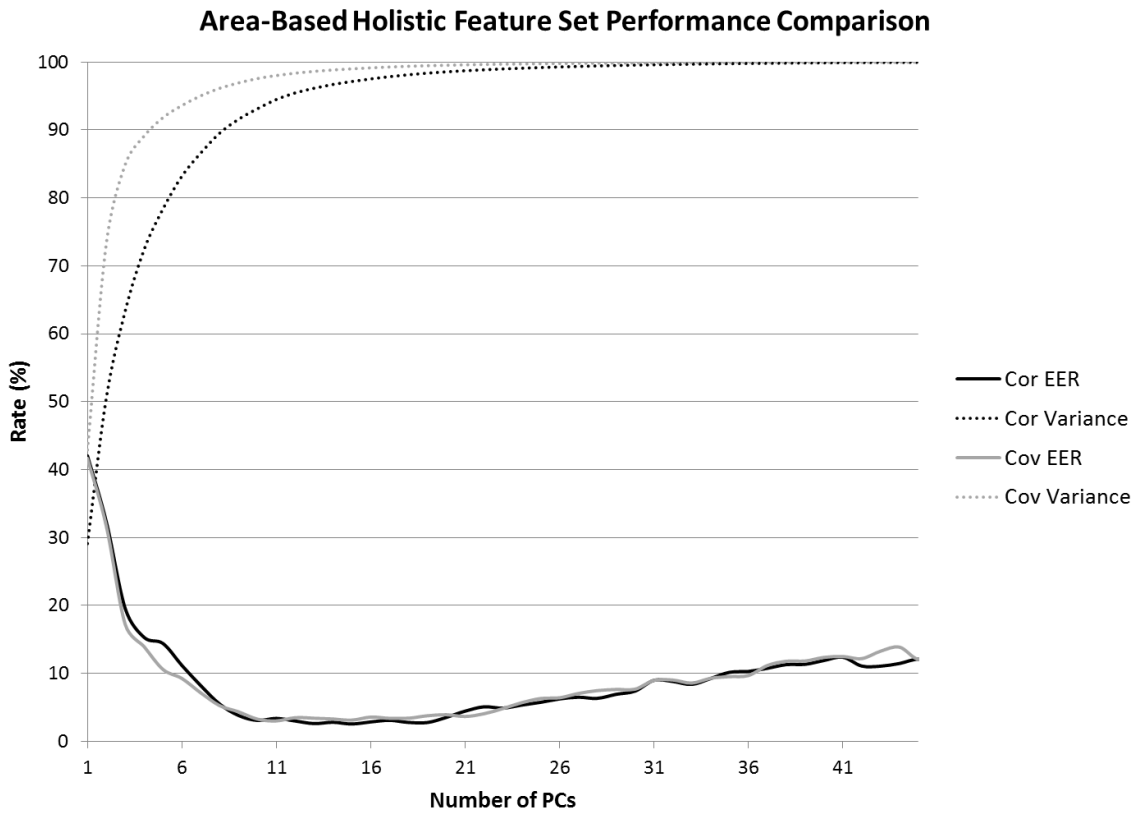

Figure 4.8: This figure demonstrates the performance of the area-based holistic approach using both covariance (cov) and correlation (cor) matrices during PCA.

To use the area-based holistic approach we first needed to decide on the number of area regions into which the original sample space should be divided. After testing the performance at 50-region intervals, from between 1000 regions per output signal and 50 regions per output signal, we found the number of regions had relatively little impact on performance and settled on 500 regions per signal for a new total sample space of 4000 area features. The results from running the area-based holistic feature extraction technique on our KNN classifier are shown in figure 4.8. As was the case when assessing the point-based feature extraction technique, the results shown here were also obtained via cross validation. In contrast to the results obtained from the point-based holistic

technique, when the area-based holistic technique was used there was a significant difference in performance between the covariance matrix and the correlation matrix-based approaches, with the covariance-trained system achieving an optimal EER of 3.0% and the correlation achieving an optimal EER of 2.55%.

Comparing the point-based holistic approach to the area-based holistic approach, we found both approaches resulted in a dimensionality reduction of about 99.8%. We also found that the area-based approach achieved better recognition performance; however, neither approach performed as well as the optimized geometric approach in the previous section. This may be a consequence of the variations in stepping speed between samples. Differences in stepping speed meant the sample space features passed for analysis via PCA were not being assessed on a single standard scale, but rather multiple scales depending on the step duration. In [3, 4], the point-based holistic feature extraction approach achieved better performance than the optimized geometric approach. Part of this may be due to the fact that these studies used a much larger training dataset than ours to generate their PC features. But these studies also used a data normalization technique to ensure the different samples were compared on the same scale. To address this issue, we explore various normalization techniques in chapter 5.

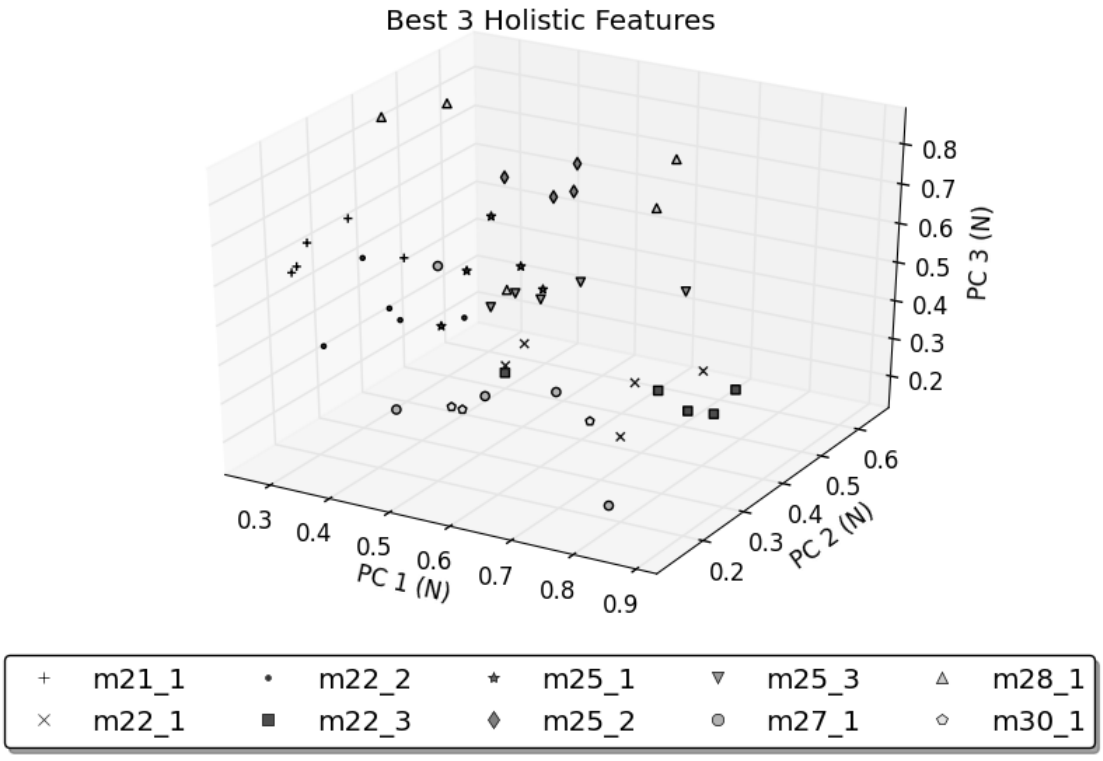

Figure 4.9: This diagram presents a visualization of the 3 best area-based holistic PC features taken from footsteps belonging to the ten test subjects and projected into a 3d frame. Five footsteps per person are shown in this diagram and the footsteps belonging to each subject are distinguished by variations in the marker symbols used. For better visualization the range for each feature has been standardized as [0,1].

Feature Space Comparison

Feature Space	Cross Validated EER (%)	Dimensions
Point-Based Holistic (covariance)	3.42222	15
Point-Based Holistic (correlation)	3.54444	16
Area-Based Holistic (covariance)	3	11
Area-Based Holistic (correlation)	2.55555	15

Table 4.4: This table compares the performance of holistic feature spaces on the development dataset.

4.3 Spectral

Important features are not always apparent in the time-domain and occasionally may become more apparent when a data sample is analyzed in the frequency domain. Two previous footstep GRF recognition studies [32, 5] have suggested techniques for extracting features from the frequency domain; we refer to these features as spectral features. In [32], Suutala and Röning proposed that features be extracted from the magnitude of the GRF and GRF-derivative frequency spectra; while in [5], Cattin proposed that features be extracted from another representation of the magnitude spectra called the Power Spectral Density (PSD), with only the derivative of the vertical GRF component examined. Both studies used PCA-based dimensionality reduction techniques to assist in the discovery of a smaller optimized spectral feature space. In our research, we decided to test both the magnitude spectra and PSD approaches, while also incorporating our PCA technique from the previous section.

To apply the two spectral approaches and accurately assess their effectiveness we extended our holistic approach to include two more steps. Prior to performing PCA, we add a filter which converts the dataset to the frequency domain then return either its magnitude spectra or PSD. Additionally, to stay consistent with the work presented in the previous GRF spectral feature studies, we have included an optional filter to obtain the GRF-derivative before the transformation to the frequency domain. And, as was the case for holistic feature extraction, the process begins by generating either a standardized point-based sample space or one based on regional areas of the GRF. The resulting

spectral features extraction process is shown in figure 4.10, while an example comparing the GRF to the GRF derivative is shown in figure 4.11.

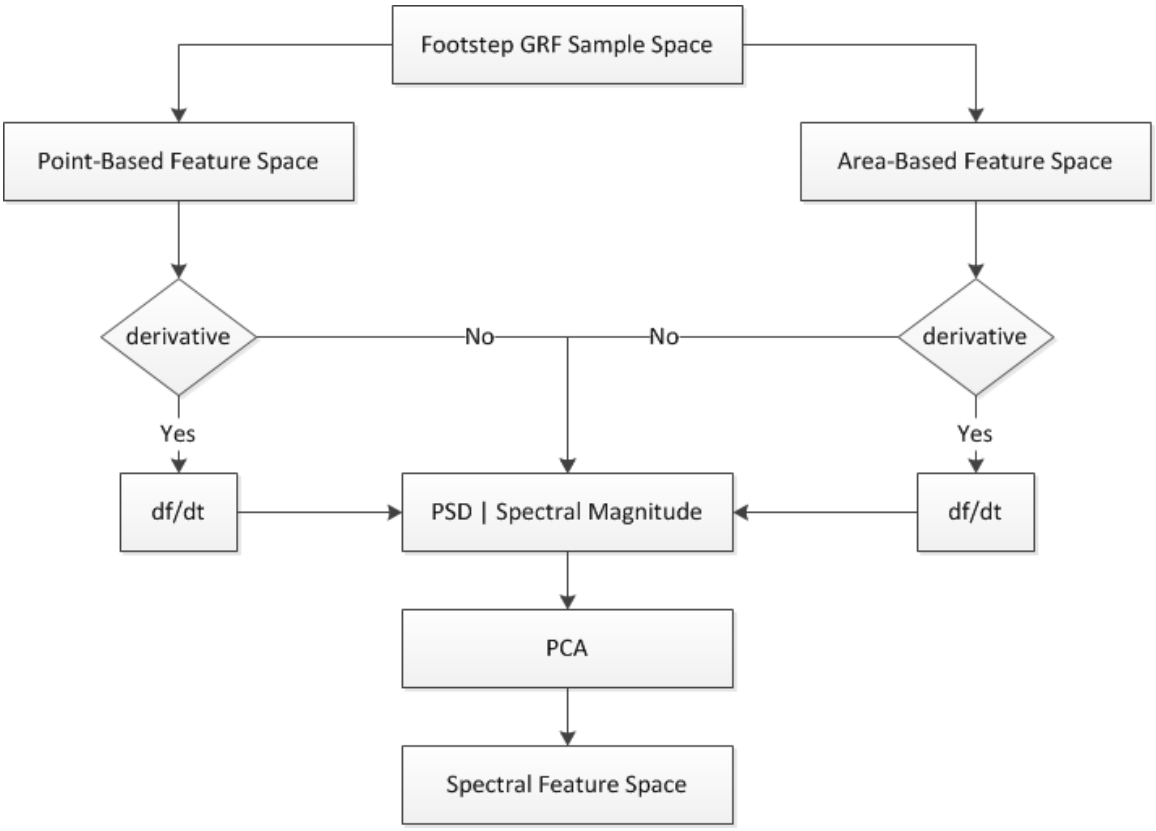

Figure 4.10: This figure represents the process used to generate the spectral feature space. Taking the derivative of the GRF before processing is optional, but the method chosen for training PCA must also be used for testing.

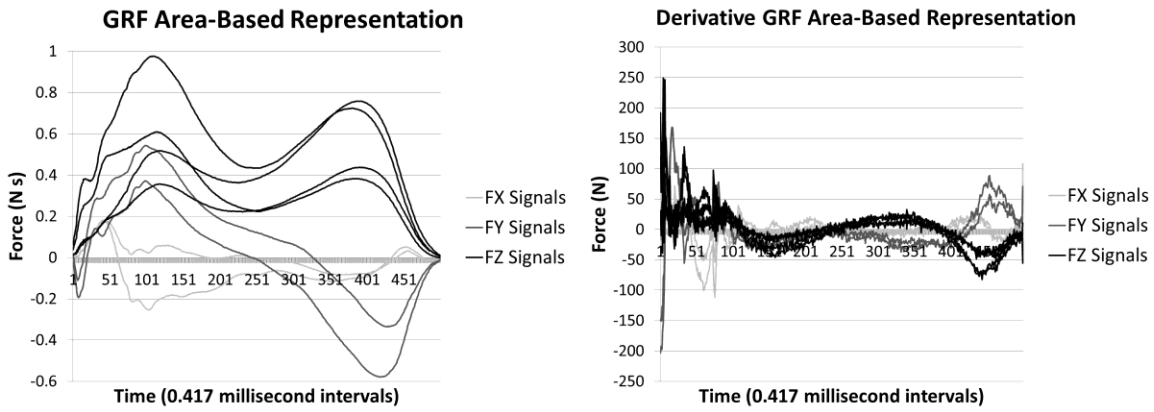

Figure 4.11: The graph on the left represents the regional area-based features in Newton-seconds at the time interval they were measured, while the graph on the right represents its derivative.

Transforming finite data series to the frequency domain is accomplished using the Discrete Fourier Transform (DFT) (equation 4.10). When this transform is performed it returns a series of complex numbers that can be processed to determine spectral phase and magnitude. We find the spectral magnitude of this series by calculating the square root of the sum of the squares for each of its real and imaginary parts (equation 4.11). Calculating the PSD is a little more challenging and requires finding an estimation called a periodogram. For the purpose of this thesis, we have implemented our periodogram and DFT using the code presented by Press et al. in Numerical Recipes [50]. In their application, they used a Fast Fourier Transform (FFT) to optimize the derivation of the frequency domain and included several optional non-rectangular window functions to counter spectral leakage during the calculation of the periodogram. Spectral leakage becomes a problem when a signal does not end at its periodic interval, which results in the unwanted "leakage" of any incomplete periodic cycles at the signal's boundary into nearby frequency bins. In our sample space, leakage was not found to be a major issue, so

in our implementation we opted for the rectangular-windowed calculation of the periodogram (equation 4.12).

$$C_k = \sum_{j=0}^{N-1} c_j e^{\frac{2\pi i j k}{N}} \qquad k = 0, \dots, N-1$$

Equation 4.10: The DFT; c_j contains the GRF or derivative GRF record at the j^{th} interval, k is the index of frequency spectral lines, and N represents our sample size.

$$|C_k| = \sqrt{Re(C_k)^2 + Im(C_k)^2}$$

Equation 4.11: The magnitude of the frequency spectrum at index k; *Re* represents C_k's real term and *Im* its complex term.

$$P(0) = P(f_c) = \frac{1}{N^2}|C_0|^2$$

$$P(f_k) = \frac{1}{N^2}[|C_k|^2 + |C_{N-k}|^2] \qquad k = 1, 2, \dots (\frac{N}{2} - 1)$$

$$P(f_c) = P\left(f_{\frac{N}{2}}\right) = \frac{1}{N^2}\left|C_{\frac{N}{2}}\right|^2$$

$$f_k \equiv \frac{k}{N\Delta} = 2f_c\frac{k}{N} \qquad k = 0, 1, \dots \frac{N}{2}$$

Equation 4.12: The periodogram estimate of the power spectrum at *N/2 + 1*, as defined by [50]. In this equation f_k is defined only for zero and positive frequencies.

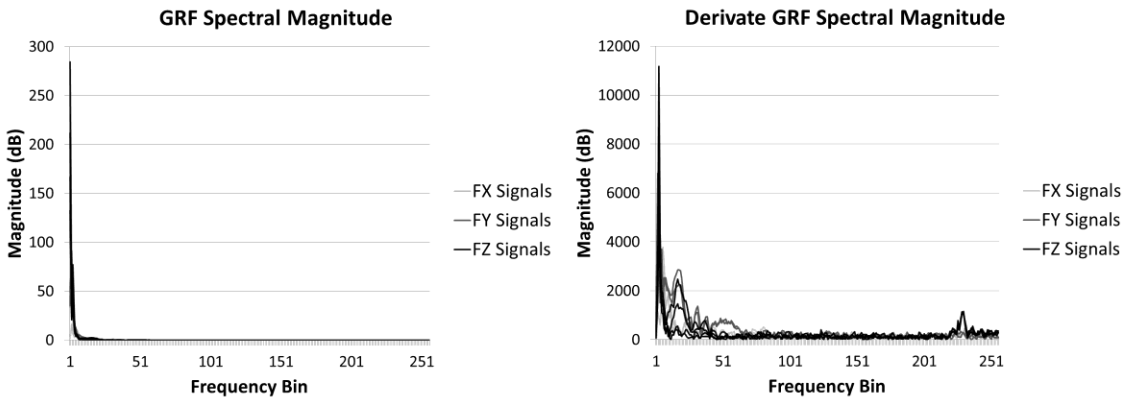

Figure 4.12: This figure provides a comparison of the spectral magnitude between the area-based GRF and area-based derivative GRF using the sample in figure 4.11.

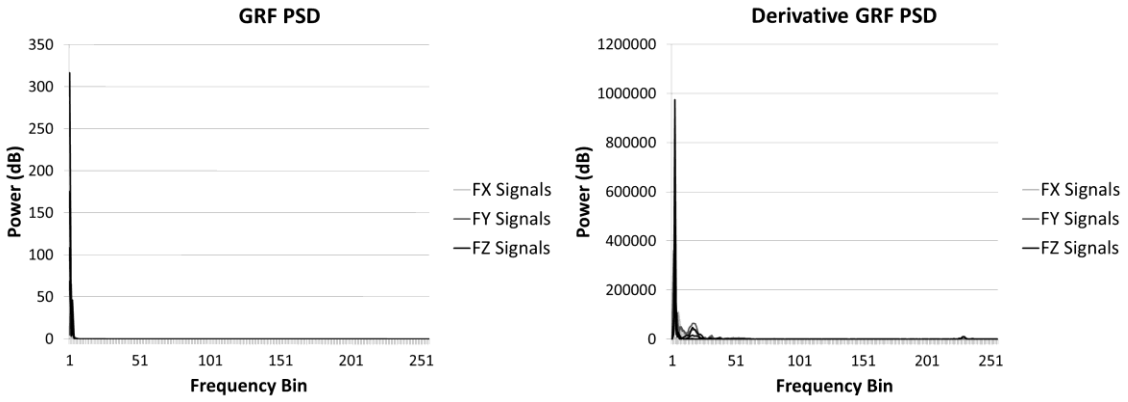

Figure 4.13: This figure provides a comparison of the PSD between the area-based GRF and area-based derivative GRF using the sample in figure 4.11.

The spectral magnitude and PSD that resulted from transforming an area-based sample footstep GRF and its derivative are shown in figures 4.12 and 4.13, respectively. As demonstrated in these figures, our spectral sample space is derived by performing the frequency domain transformation separately on each output signal rather than on any combination of signals; however, when PCA is performed, all 8 extracted frequency domains are concatenated into a single space for analysis. The above frequency domain representations showed little in the way of visual characteristics since they tended to be

heavily dominated by the strength of a small number of lower frequency bins. Yet the performance of these spectral feature sets, shown in the figures below, turned out to be similar to the performance achieved using the holistic extraction technique.

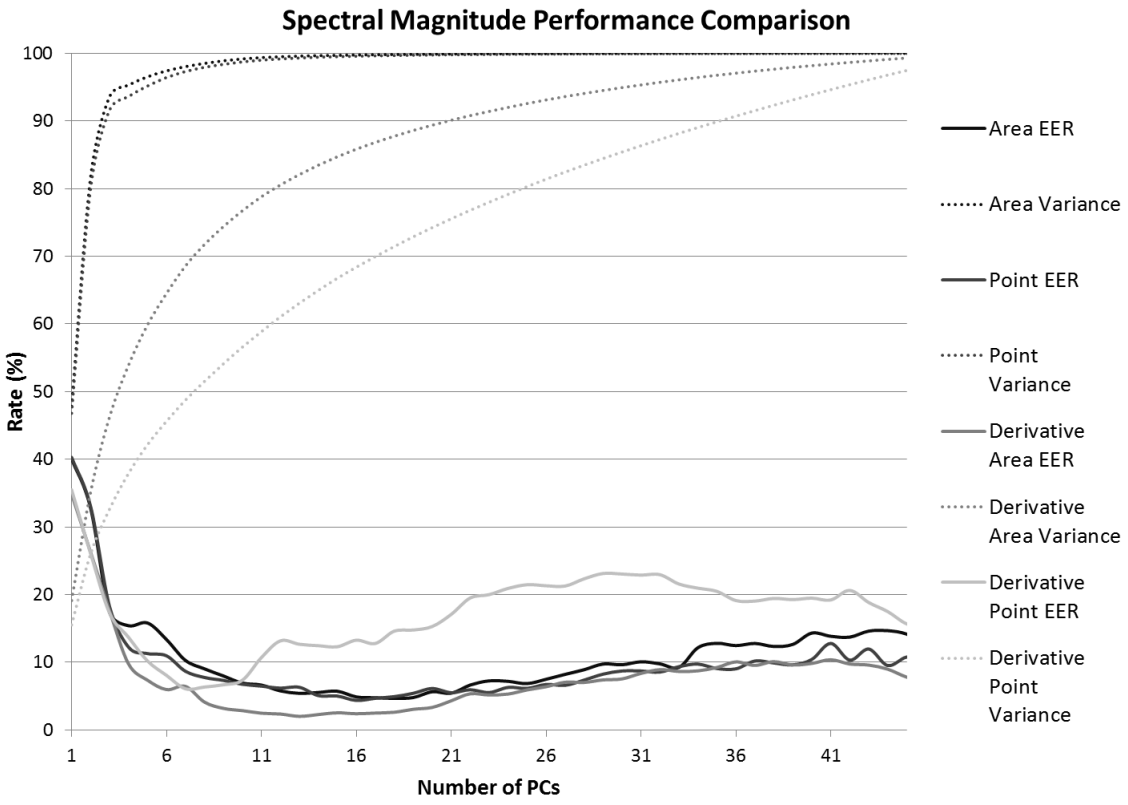

Figure 4.14: This figure compares the performance of our four spectral magnitude feature spaces.

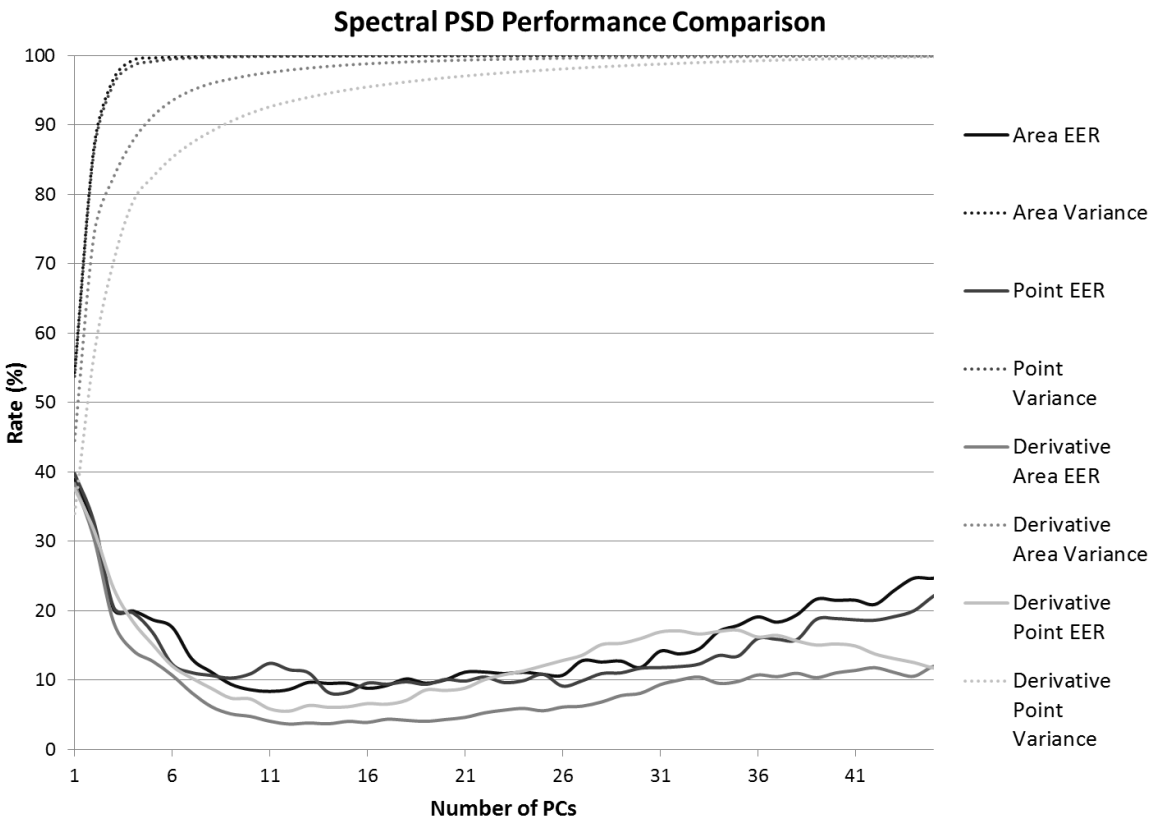

Figure 4.15: This figure compares the performance of our four spectral PSD feature spaces.

The combination of the area/point-based approaches and derivative/non-derivative representations gave us four different spectral feature spaces to analyze. The results from performing classification on our development dataset with these four spectral feature spaces in terms of both spectral magnitude and PSD are shown in figures 4.14 and 4.15. All demonstrated results were achieved using the covariance PCA configuration, which performed better than the correlation configuration. And, once again, we used our KNN classifier to acquire the EER, with an area-based space composed of 500 area regions per output signal and point-based space composed of 2000 points per output signal. In both the spectral magnitude and PSD feature spaces, the best performance came from the area-based GRF derivative features. Furthermore, while each extractor reduced feature space

dimensionality by 99%, the spectral magnitude feature spaces clearly performed better than the PSD feature spaces, with an optimal spectral magnitude EER of 2.02% versus an optimal PSD EER of 3.68%. However, it should be noted that while our analysis found that features extracted from the spectral magnitude sample space performed better than those extracted from the PSD feature space for a single footstep, a direct comparison could not be made with the PSD method used in [5] since the work in that paper performed recognition using a multi-footstep sample space and a generalized variation of PCA.

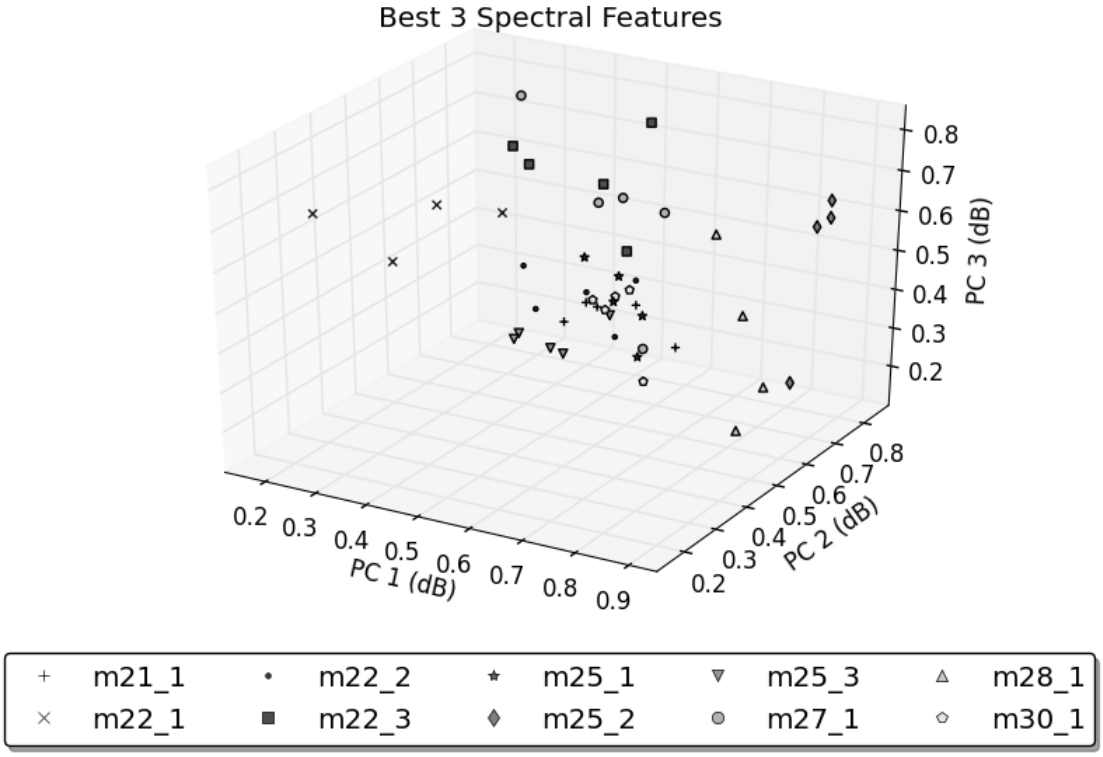

Figure 4.16: This diagram presents a visualization of the spectral magnitude PC features, obtained using the area-based derivative sample space taken from footsteps belonging to the ten test subjects and projected into a 3d frame. Five footsteps per person are shown in this diagram and the footsteps belonging to each subject are distinguished by variations in the

marker symbols used. For better visualization the range for each feature has been standardized as [0,1].

Feature Space Comparison

Feature Space	Cross Validated EER (%)	Dimensions
Point-Based Spectral Magnitude	4.37777	16
Point-Based Derivative Spectral Magnitude	6.1	7
Area-Based Spectral Magnitude	4.67777	18
Area-Based Derivative Spectral Magnitude	2.02222	13
Point-Based Spectral PSD	9.4	19
Point-Based Derivative Spectral PSD	5.55555	12
Area-Based Spectral PSD	8.38888	11
Area-Based Derivative Spectral PSD	3.68888	12

Table 4.5: This table compares the performance of spectral feature spaces on the development dataset.

4.4 Wavelet Packet

In the previous section we explored the idea that important GRF characteristics may be found in the frequency domain. Analysis of frequency given a time domain-based dataset has traditionally been done via the application of the Fourier transform (the approach used in our spectral feature extractors). However, when the Fourier transform is applied, significant information regarding the location of particular frequencies will be lost [51]. In [7], Moustakidis et al. proposed an alternative form of GRF frequency analysis based on the Wavelet Packet (WP) transform. While the domain obtained by the Fourier transform is characterized by basis functions consisting of sine and cosine functions, the domain obtained by the WP transform is characterized by basis functions that are localized over a finite space and called wavelets. This space-localization property of the WP transform makes it possible to effectively analyze frequencies that occur over a particular period of time; consequently, the domain resulting from the WP transform is often referred to as the time-frequency domain. In the research presented in this thesis we refer to the features extracted in the WP time-frequency domain as wavelet features, and we have based our analysis of these features on the work done in [7, 52].

The feature extraction technique described in [7] was based on a proposal by Li et al. [52] to improve the classification of biomedical signals. In their proposal they outlined a two stage process for extracting features; the first stage involved performing a WP transform, while the second stage used fuzzy sets to identify the most discriminant features in the new WP space. The application of fuzzy set-based feature identification technique was significant. Unlike the PCA-based approaches, which identified important

characteristics with no prior knowledge of the subjects used in training, the fuzzy set technique was made fully aware of the subjects attached to each training sample and used this knowledge to construct a ranking of discriminative features. In machine learning these class-aware algorithms are known as supervised learning models, and, by including this approach in our research, we were able to compare and contrast its performance with that of the unsupervised PCA approach; albeit, the comparison is done across differing domains.

As mentioned in the previous paragraph, the fuzzy WP feature extraction technique is divided into two stages. The first stage involves performing Wavelet Packet Decomposition (WPD) on the sample space for each of our training samples. During WPD, each studied sample is passed through a filter bank defined by a chosen wavelet function; this filter bank consists of a high and low pass filter, and divides the sample space along the center of its frequency spectrum producing one subspace representing the upper half of the original sample space's frequency spectrum and the other representing its lower half. Next the resulting frequency subspaces will each be passed back through the wavelet filter bank producing four subspaces, and this process of further dividing the frequency subspaces will continue until a specified level of decomposition is completed or the Nyquist limit for the sample space is reached. The end result of the WPD can be represented as a tree with exactly two nodes per branch.

In the second stage of the fuzzy WP feature extraction technique, we search the WPD tree for its most discriminative characteristics. To do this we first want to find the most

discriminative set of WPD nodes covering the entire sample space frequency spectrum such that there is no overlap between the spectra of individual nodes. This is done to ensure no redundant information is used in analysis and this representation is called the optimal WPD. To find the optimal WPD we need to rank each WPD node for its discriminative ability. In [52] node ranking was accomplished using a function based on fuzzy c-means clustering (equation 4.13). For each WPD node, this function examines all training samples and determines a degree to which the WPD coefficients in the given node correspond to their mean value for the subject they represent; these values are summed up for all coefficients in the node and the higher the resulting total, the better the discriminative ranking assigned to the given node. Unfortunately, this process can suffer when the dataset contains coefficients that produce poor degrees of membership; in this case a single poor coefficient or small group of poor coefficients may contribute to a significant reduction in node ranking for nodes that contain strong discriminators and otherwise would rank strongly. To counter this condition, Li's team proposed the application of exclusion criterion (equation 4.14) to remove samples with poor discriminative ability prior to calculating the optimal WPD. Additionally, to prevent any single coefficient from having an undesirably large or small impact on the node rankings Li's team normalized each coefficient by its standard deviation.

$$u_{ik} = \left[\sum_{j=1}^{c} \left(\|\boldsymbol{x}'_{\boldsymbol{k}} - \boldsymbol{v}'_{\boldsymbol{i}}\|^2_{\sigma'} / \|\boldsymbol{x}'_{\boldsymbol{k}} - \boldsymbol{v}'_{\boldsymbol{j}}\|^2_{\sigma'} \right)^{\frac{1}{b-1}}\right]^{-1}$$

$$F(X) = \sum_{i=1}^{c} \sum_{k \in A_i} u_{ik}$$

Equation 4.131: This equation demonstrates the fuzzy membership criterion (*F(X)*) used to rank nodes in the WPD tree, where *X* is the feature space for a single node, *c* represents the

number of classes (subjects), and A_i represents the set of training data samples belonging to class *i*. Numerical scores are generated by u_{ik}, the fuzzy c-means objective function, which determines the degree to which the wavelet coefficient vector for training sample *k* (x'_k) in node *X* belongs to class *i*. In this function the prime symbol represents vectors reduced using the exclusion criterion (equation 4.14) and σ represents the normalization of coefficients by their standard deviation; v'_i represents the vector containing the wavelet coefficient means of class *i*, and *b* the fuzzifier that modifies the shape of the membership function. In assessing the fuzzy WP feature extraction technique we set $b = 2$, and applied the two boundary conditions identified in [52]: if $x'_k = v'_i$, then $u_{ik} = 1$, and, if $x'_k = v'_j$, $i \neq j$, then $u_{ik} = 0$.

$$D(j) = \frac{max\{v_{ij}|_{i=1}^{M}\} - min\{v_{ij}|_{i=1}^{M}\}}{2\sigma_j} < r$$

Equation 4.14: For a given feature *j*, the exclusion criterion (*D(j)*) is calculated taking the maximum distance in mean values (v_{ij}) for the given feature and class $i = 1 \ldots M$, dividing it by twice the standard deviation of the given feature, then comparing it against the retention threshold *r*. If $D(j) < r$, then feature *j* is excluded from the feature space.

For our research, we have used the optimal WPD technique described above, but also included the additional step of transforming the original sample space into either a dimensionally standardized point-based or area-based sample space (see section 4.2) prior to performing the WPD. We have also differentiated our GRF wavelet feature research from the work done in [7] by performing the WPD with two previously untested wavelet functions, Legendre 04 (lege04) and Legendre 06 (lege06); the performance of these two was compared with the two best wavelet functions in [7], the Coiflet 06 (coif06) and Daubechies 04 (daub04). Figure 4.17 illustrates an optimal WPD for the F1Z4 signal, calculated using the area-based dimension standardization approach with a coif06 wavelet function.

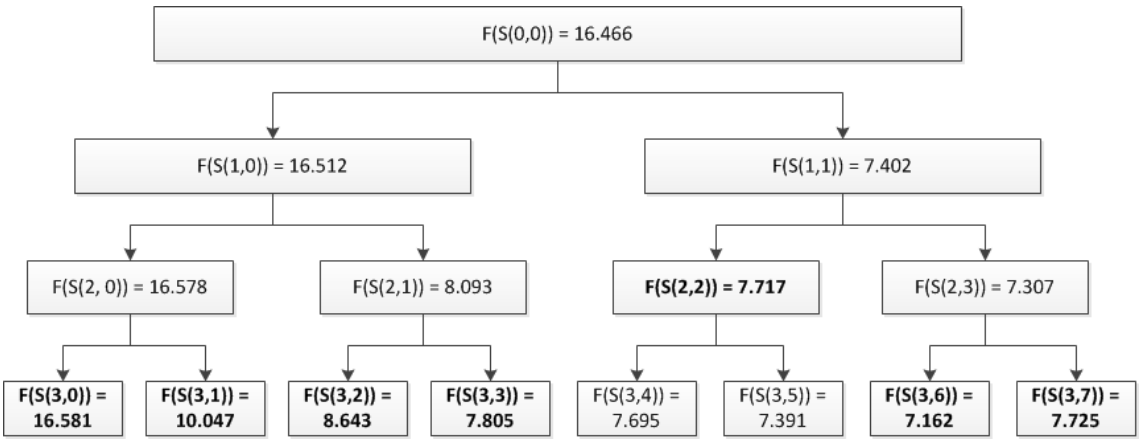

Figure 4.17: The optimal WPD tree for the F1Z4 output signal. Subspaces are represented as *S(j, k)*, where *j* represents the decomposition level and *k* spectral position of the node. The score ranking for each node is shown and the optimal decomposition is highlighted in bold.

Finding the optimal WPD gives us the wavelet feature space, but the application of dimensionality reduction to this space requires an additional step. In [7, 52], dimensionality reduction was accomplished by calculating the fuzzy membership (equation 4.15) for each individual wavelet feature, sorting the feature indices by the membership values returned, then forming the dimensionally-reduced feature set as the first *N* indices, where *N* is less than the total number of dimensions. The membership function reflects on the discriminatory power for each feature and therefore features at the start of the sorted list should be the best suited for classification. To find the best set of features for classification the choice of *N* can be optimized in the same way the number of PCs for holistic features was optimized, by plotting the size of the feature set against the EER it produces.

$$u_{ij}(x_{kj}) = \left[\sum_{m=1}^{c} \frac{(x_{kj} - v_{ij})^2}{(x_{kj} - v_{mj})^2}\right]^{-1}$$

$$F(j) = \sum_{i=1}^{c} \sum_{k \in A_i} u_{ij}(x_{kj})$$

Equation 2: This equation demonstrates the fuzzy membership feature ranking score (*F(j)*) for feature *j*. In this equation *i* represents the class, *c* the total number of classes, A_i the training samples belonging to class *i*, x_{kj} the feature *j* in sample *k*, v_{ij} the mean for feature *j* in class *i*, and u_{ij} the fuzzy membership score for feature *j* in class *i*.

When applying the described WP feature extraction technique to GRF data, Moustakidis's team calculated the optimal WPD for each of the GRF components separately, then performed feature ranking on the space derived from combining every optimal WPD. We used a similar approach in our wavelet feature extractor, but rather than calculating the optimal WPD separately for each GRF component we calculated it separately for each of our 8 GRF output signals. Using this approach, the incorporation of WP feature extraction technique into person classification was achieved through the following steps:

1) Standardize the dimensionality of each subset using the area-based or point-based approaches in section 4.2. Then compute the WPD for each output signal in the training data subset.
2) Use the WPDs calculated in step 1, together with equations 4.13 and 4.14, to calculate the optimal WPD; then transform the output signals of each training data sample into their respective optimal WPD space.

3) Combine the optimal WPDs for each output signal to form the full wavelet feature space and then use equation 4.15 to apply a fuzzy membership rank to each feature.
4) Sort the features by rank and take the first *N* best features to form the new dimensionally-reduced wavelet feature space.
5) Transform each training data sample into the reduced wavelet feature space.
6) Train the classifier using the transformed training samples.
7) Transform a data sample from the testing subset into the reduced wavelet feature space.
8) Perform classification on the transformed testing data sample using the classifier from step 6.

To implement the WP feature extractor, we first converted Christian Scheiblich's JWave [53] into C# and used it to perform the WPD, then integrated it with our own fuzzy membership-based C# solution for finding the optimal WPD and wavelet features. To find the reduction in dimensionality that best optimizes the WP feature extractor's performance, we plotted the 100 best features for various extractor configurations against the EER produced as each feature was successively included in the feature space; optimal feature sets larger than 100 dimensions were not considered competitive with the alternative feature extraction techniques and thus ignored. Once again, classification results were measured using our KNN classifier and cross validation was performed to improve accuracy. To remove bias, in each cross validation iteration a new WP feature extractor was generated. Our tested configurations included four wavelet functions

(coif06, daub04, lege04, and lege06), the wavelet decomposition depth ($L = 4$) and retention threshold ($r = 0.3$) used in [7], as well as both our point and area-based dimension standardization approaches. The point-based standardization (2048 dimensions) and area-based standardization (512 dimensions) were set as powers of 2 to facilitate any level of decomposition. The results from running these wavelet packet extractor configurations are shown in figure 4.18.

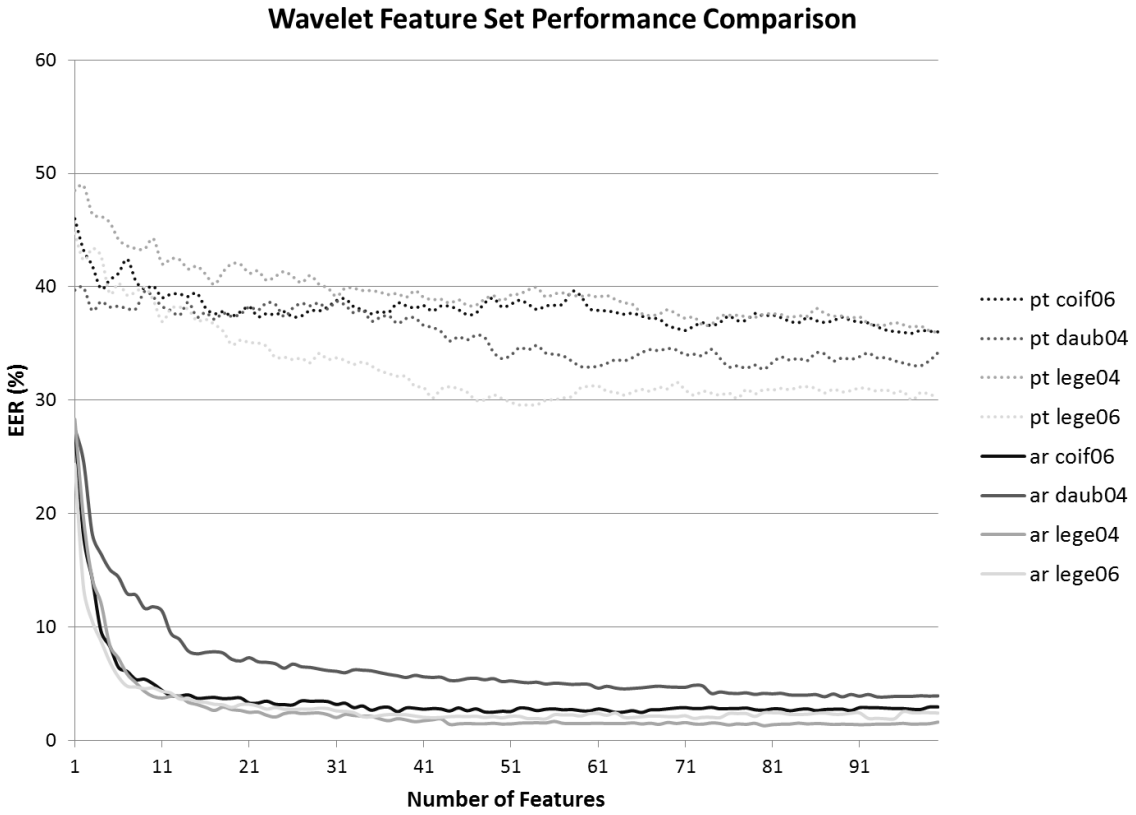

Figure 4.18: This figure demonstrates a comparison of the performance between different wavelet feature extractor configurations. In this diagram 'pt' represents point-based approaches while 'ar' represents area-based approaches.

Analyzing the results in figure 4.18, it is apparent that the WP feature extractor performed much worse when run on the point-based sample spaces. This may be a

consequence of the differing sample space sizes; while the point-based approach produced a sample space of 16384 dimensions for reduction, the area-based approach consisted of only 4096 dimensions. Given such a large number of point-based dimensions, the extractor may have assigned a high ranking to a number of features that performed well individually, but shared redundant information with their high ranking peers, and contributed to no increase in performance when grouped into a feature set. Another possibility may be that the WP feature extraction technique is not well suited for identifying GRF features when substantial differences in feature space alignment are present, as was the case for the point-based approaches. In [7], samples were discretized into 700 dimensions per GRF component, producing a total of 2100 dimensions and a proportional sample space similar to the one produced by our area-based approach. Classification performance in [7] was relatively close to the performance achieved using our area-based WP feature extractor, yet, unlike the results of [7], our best performance came when WPD was run using the Legendre 04 wavelet function.

Using the area-based WP feature extractor with the Legendre 04 wavelet function, we achieved a best EER of 1.28% for a feature set of 80 dimensions (a 99.3% decrease in dimensionality). One interesting by-product of the WP feature extractor was the grouping of features according to their corresponding output signals. This is demonstrated in figure 4.19, where the 80 best features are labeled with their output signal and measured by their fuzzy membership ranking and position in the wavelet feature space. We discovered that, in the first cross validation of our development dataset, 45 of our best wavelet features were derived from the anterior-posterior GRF component, 14 were derived from the

vertical GRF component, and 21 were derived from the medial-lateral GRF component. This finding was particularly interesting, because it contradicted the conclusion in [7] that the GRF vertical component is best suited for subject recognition.

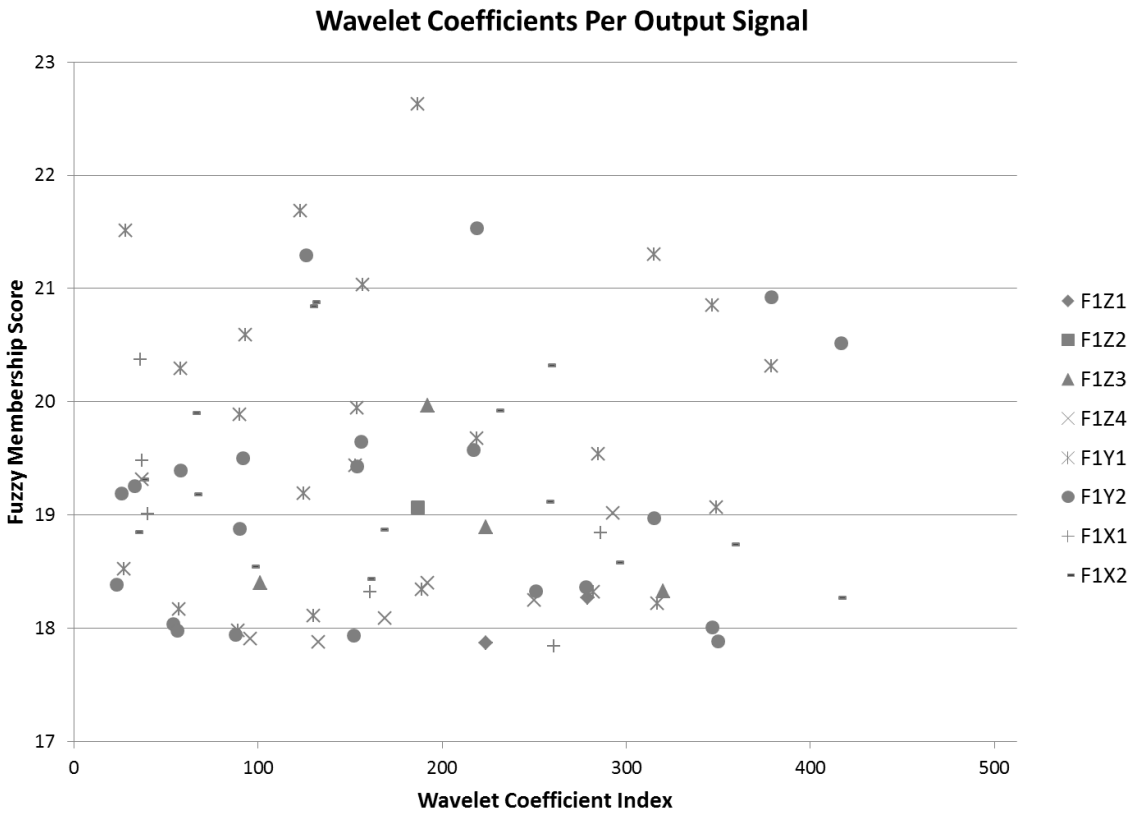

Figure 4.19: This figure demonstrate the 80 best features derived using the area-based dimensionality standardization approach with a WP feature extractor based on the lege04 wavelet function. The fuzzy membership score in the y-axis refers to the feature ranking scores returned by equation 4.15, while the wavelet coefficient index refers to the index of the feature in the optimal WPD feature space.

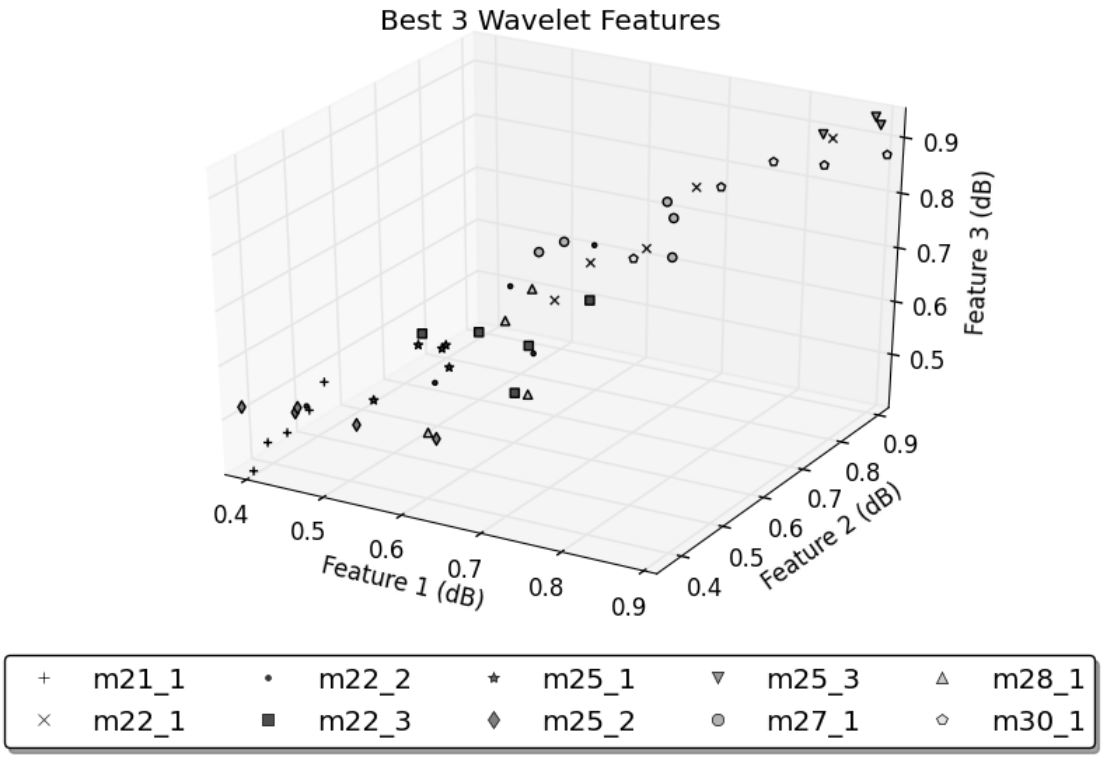

Figure 4.20: This diagram presents a visualization of the 3 best wavelet features, obtained using the area-based sample space taken from footsteps belonging to the ten test subjects and projected into a 3d frame. Five footsteps per person are shown in this diagram and the footsteps belonging to each subject are distinguished by variations in the marker symbols used. For better visualization the range for each feature has been standardized as [0,1].

Feature Space Comparison

Feature Space	Cross Validated EER (%)	Dimensions
Point-Based Coif06	35.9	97
Point-Based Daub04	32.72222	80
Point-Based Lege04	35.98888	100
Point-Based Lege06	29.57777	52
Area-Based Coif06	2.45555	66
Area-Based Daub04	3.82222	94
Area-Based Lege04	1.28888	80
Area-Based Lege06	1.88888	55

Table 4.6: This table compares the performance of wavelet feature spaces on the development dataset.

4.5 Summary

This chapter presented the concept known as feature extraction as a means to transform large noisy data sample spaces into smaller useful feature spaces, ideally retaining only the information most relevant to the data analysis objective. In our case the underlying objective was to extract the features from the footstep GRF best able to discriminate one individual's GRF from another's, and, to achieve this objective, we analyzed and implemented four different feature extraction techniques. In our research the terms "feature" and "dimension" were used interchangeably while the process of reducing the size of the feature space was often referred to as dimensionality reduction. Furthermore, it was shown that feature spaces can be described as being either heuristically selected or discovered via machine learning techniques. In our research both methods for establishing feature spaces were presented, with the geometric space being heuristically defined, and the holistic, spectral and wavelet spaces defined via machine learning.

To optimize each feature extractor so that they best conformed to our GRF data we performed an optimal value search by calculating and comparing the EER for varying configurations. In each case the EER was calculated using the KNN classifier from chapter 6 with the value of K set to 5. After running these results for each feature extractor we found the wavelet feature space to be the most performant followed by the geometric, spectral and lastly the holistic space. In this chapter we distinguished machine learning feature extractors as being either supervised or unsupervised, an important distinction when interpreting the results. In this case the wavelet extractor was shown to be supervised and as such may have benefited from a positive bias in its results due to its

underlying exposure to the boundaries between subject samples. Moreover, the geometric feature extractor, while not a machine learning extractor in direct feature discovery, could also be considered supervised in that it used a supervised brute force optimization approach to limit the geometric features applied only to those producing the best performance. Conversely, the holistic and spectral feature extractors were developed with no understanding of the underlying subject divisions and thus could be considered unsupervised. Our top results for each of the aforementioned feature spaces are demonstrated in table 4.7. In the next chapter we explore the use of dataset normalization as a means to improve upon these results by assisting the feature extractors in selecting features better able to differentiate GRF subjects. The next chapter places a particular emphasis on finding and exploiting the relationship between stepping speed and GRF curve signature to help assess the second of our problem statement assertions.

Feature Space GRF Recognition Performance

Feature Space	EER (%)
Optimal Geometric	1.33333
Best Holistic	2.55555
Best Spectral	2.02222
Best Wavelet	1.28888

Table 4.7: This table compares the best GRF recognition performance achieved across feature spaces extracted using various extraction techniques.

Chapter 5

Normalization

Using a feature extraction technique can assist in the discovery of discriminant features, and, in datasets containing sources of intra-subject sample variability, feature extraction techniques may identify discriminant features not affected by such variability. However, when it is possible to identify these sources of variability it may also be possible to use normalization to expose important features that would otherwise be hidden due to differences in the conditions at the time of sample collection. To determine the variance that can be accounted for by normalization we must find the sample space attributes that both appear consistently across the sample space and correlate to the conditions experienced during data capture. With regards to footstep GRF recognition, three previous types of inter-subject sample variability have been used for normalization in existing studies: the observed GRF curve amplitude [3, 4], the step duration (the length of the time the foot is on the ground during a step) [7], and the weight of the studied subject [7].

For the purpose of our research, we have based our recognition model around the assumption that nothing is known about either the subjects or conditions experienced during data collection, leaving only the GRF signature to analyze. In the absence of additional information regarding the conditions experienced during sample collection, normalization could still be accomplished by scaling and/or shifting the GRF force

signatures such that they line up according to some standard set of graphical and/or statistical data properties. Alternatively, for the GRF, normalization could also be accomplished by modeling the relationship between step duration and the GRF force curve then transforming the location of each feature to the location it would be expected to be located at were sample space step durations aligned. The normalization research presented in this chapter distinguishes our work from previous studies, in that we are, to our knowledge, the first to do an in depth analysis on the impact of step duration model-based normalization on GRF recognition. In the following sections we begin our analysis by examining normalization based on simple traditional linear scaling and shifting operations, and then we introduce two new normalization techniques built around the modeling of the relationship between step duration and the shape of the GRF force curve. To demonstrate the impact that each normalization technique had on GRF recognition we normalized our development dataset with every normalizer and passed the results to our best feature extractors, again using the simple KNN classifier produce our recognition results.

5.1 Scaling and Shifting

The simplest category of normalization involves the application of a single scale and/or shift operation to transform all samples in a dataset to a chosen common scale. Ideally, after each sample has been transformed to the new scale, all intra-subject variability would be removed with only the inter-subject variability remaining; this would allow for perfect subject recognition. For instance, if the GRF force signature shape was unique for each subject, but the amplitude of the signatures in the dataset varied with respect to a constant across samples, then by scaling the samples such that each was standardized to a common maximum amplitude, all intra-subject variance would be removed exposing the remaining inter-subject variance. While it is very unrealistic to expect such an ideal scenario in highly variable data like the GRF force signature, these techniques can still often result in some degree of variance reduction. For the purpose of our research, we examined the impact of five such normalizers on GRF recognition performance: the L^{∞} normalizer, the L^{1}-normalizer, the L^{2}-normalizer, Linear Time Normalization, and Score normalization.

The L^{∞}, L^{1}, and L^{2} normalizers are the most basic of our chosen normalizers. Using these techniques, normalization is accomplished by scaling each GRF force signature in our dataset via the inverse of either its L^{∞} norm (equation 5.1), L^{1} norm (equation 5.2), or L^{2}-norm (equation 5.3), respectively. Of these three, only the L^{∞} normalization technique has been previously used to normalize the footstep GRF for recognition purposes; however, the L^{1} and L^{2} normalization techniques have been used in a number of image-based recognition studies [54, 55, 56], and, in [57], it was found that, by performing L^{1}

or L^2 normalization after PCA compression, facial recognition rates increased by as much as 5%. Our application of these three normalizers used the combined feature space for all 8 GRF force signals to calculate their norms. In the case of the optimal geometric feature space, the optimal features were recalculated using the normalized data and normalization was only applied to the force values with no alterations made to statistical or temporal feature values. The results achieved after applying these normalizers to our development dataset and best performing feature extractors are demonstrated in table 5.1. Both the L^1 and L^2 normalizers led to a significant increase in GRF recognition performance over the equivalent non-normalized results when combined with our best holistic feature extractor; however, all other uses of the L-type normalizers demonstrated a large reduction in recognition performance.

L^∞ Normalized Feature Extractors

Feature Extractor	Cross Validated EER (%)	Dimensions	EER Improvement (%)
Optimal Geometric	2.14444	27	-60.8
Best Holistic	3.26666	13	-27.8
Best Spectral	3.55555	16	-75.8
Best Wavelet	2.71111	100	-52.4

L1 Normalized Feature Extractors

Feature Extractor	Cross Validated EER (%)	Dimensions	EER Improvement (%)
Optimal Geometric	1.46666	47	-9.9
Best Holistic	2.04444	13	20
Best Spectral	2.87777	15	-42.3
Best Wavelet	1.63333	97	-26.7

L2 Normalized Feature Extractors

Feature Extractor	Cross Validated EER (%)	Dimensions	EER Improvement (%)
Optimal Geometric	1.48888	69	-11.6
Best Holistic	2.05555	16	19.5
Best Spectral	2.74444	15	-35.7
Best Wavelet	1.56666	76	-21.5

Table 5.1: This table demonstrates the change in performance achieved by the L-type normalizers against the best performing feature extractors from chapter 4.

$$\|y\|_\infty = \max_{i \in N}(|y_i|)$$

Equation 5.1: This equation presents the L^∞ norm, where *N* is the data sample length. Scaling the dataset by the L^∞ norm will result in every data sample having maximum absolute amplitude of 1.

$$\|y\|_1 = \sum_{i \in N}|y_i|$$

Equation 5.2: This equation presents the L^1 norm, also known as the Manhattan norm, where *N* is the data sample length. Scaling the dataset by the L^1 norm will result in every data sample having a unit area of 1.

$$\|y\|_2 = \sqrt{\sum_{i \in N} y_i^2}$$

Equation 5.3: This equation presents the L^2 norm, also known as the Euclidean norm, where *N* is the data sample length. Scaling the dataset by the L^2 norm will result in every data sample having a unit length of 1.

While the L-type normalizers involved aligning data samples according to their amplitude-based norms, data can also be normalized with respect to its temporal properties. One method for performing temporal normalization, previously used for footstep GRF recognition in [7], is Linear Time Normalization (LTN). LTN refers to the process of linearly aligning the phase of data samples with differing durations into a standard reference frame such that the samples' proportionate temporal properties can be directly compared. Its performance depends on the assumption that signal duration bears no influence on signal amplitude. Figure 5.1 demonstrates an ideal example of the transform performed by LTN to temporally align two different-duration data samples

from the same subject. Under the ideal scenario, performing LTN will result in the phase and amplitude of any same-subject samples lining up at a position different from that of any other subject.

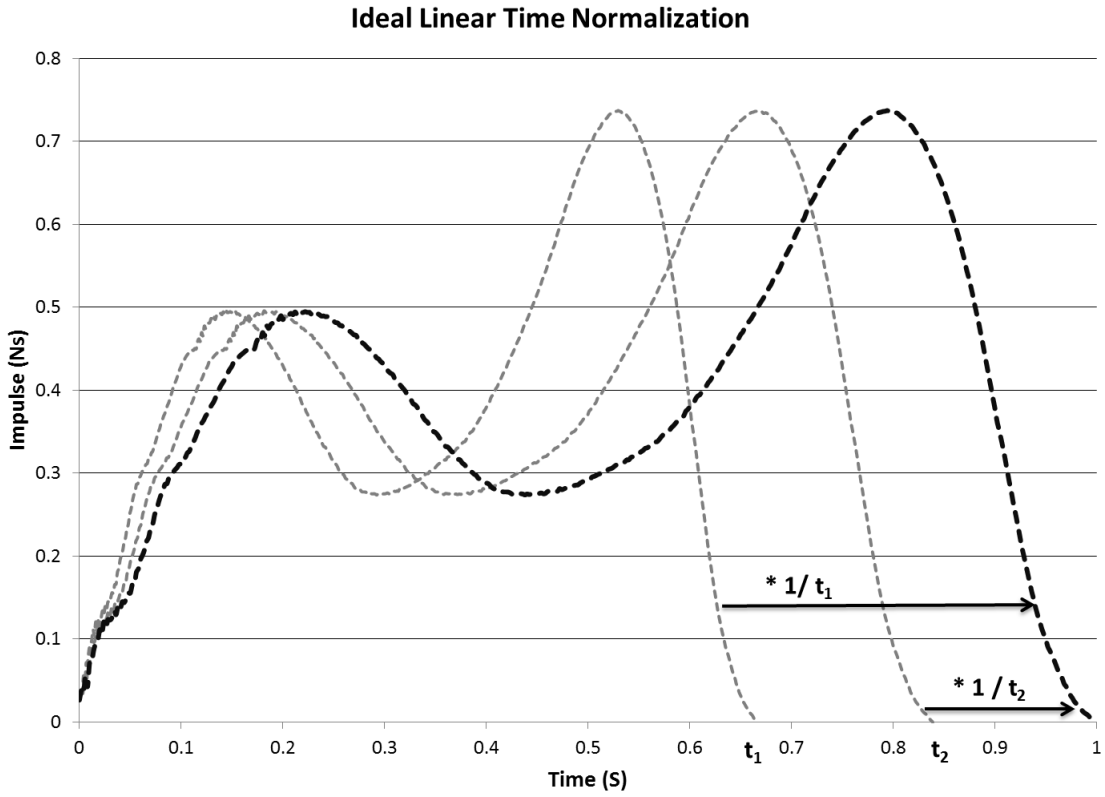

Figure 5.1: This figure demonstrates the scaling of the samples ending at t_1 and t_2 by LTN to a common length of 1. Under the ideal scenario shown here the samples line up perfectly when compared in the same phase.

The method of implementation used to perform LTN depends on the type of feature set being normalized. If the features to be normalized have a full or partial temporal component, LTN can be accomplished via scaling them by the ratio of a chosen standardized signal duration (i.e. 1 second) to the duration of the signal being normalized (i.e. 0.71 seconds). Whereas, if the features to be normalized represent the amplitude

records of varying length time series recorded at a standard sampling rate, then LTN can be accomplished by resampling each signal to a standard number of records [58]. In our analysis, this meant using the ratio-based scaling LTN method to normalize the time and area features in our geometric extractor, and the resampling LTN method to normalize the input to our machine learning-based feature extractors. As the area-based sample representation used on the input for all of our best feature extractors already performed resampling in its derivation, to accomplish LTN for our non-geometric feature spaces we simply took the set of aligned areas produced by the approach and re-scaled each sample so to remove the time component from each of the representation's area features.

The results of LTN on the applicable normalizers are shown in table 5.2 below. LTN was not applicable for the spectral derivative magnitude feature set, as taking the derivative of the set already negated the time dimension. Looking at our results, the LTN technique achieved better recognition performance for each of the applicable feature extractors. This appears to suggest that, by removing the influence of step duration via normalization, we can increase GRF recognition performance.

LTN Normalized Feature Extractors

Feature Extractor	Cross Validated EER (%)	Dimensions	EER Improvement (%)
Optimal Geometric	1.03333	46	22.5
Best Holistic	2.3	16	9.9
Best Wavelet	1.1	97	14.6

Table 5.2: This table demonstrates the change in performance achieved by the LTN normalizer against the best performing feature extractors from chapter 4.

An alternative to the previously described scaling normalizers is a category of normalization known as Score normalization. These normalizers are widely used in statistics and have also found their way into biometric applications such as speech recognition [59]. For the purpose of our research we have applied the standard score (or Z-score) normalizer (equation 5.4) to our development dataset. This normalizer shifts and scales the data such that every sample in the dataset will have a mean value of zero and standard deviation of one. The Z-score normalizer was previously applied to the training samples in the calculation of the PCA correlation matrix for our holistic feature extraction technique in chapter 4; however, in this section the feature extractor is left unaware of the normalization and the Z-score normalizer is applied to all data samples prior to its derivation. The result obtained after applying the Z-score to our development dataset, shown in table 5.3, showed a slight decrease in recognition performance when the normalizer was applied to the holistic area-based feature extractor and a substantial decrease in recognition performance for all other feature extractors.

$$x_{i \in N} = \frac{x_i - \mu}{\sigma}$$

Equation 5.4: This equation demonstrates how the Z-score is applied to each feature x_i in a sample of length N with a mean of μ and standard deviation of σ.

Z-Score Normalized Feature Extractors

Feature Extractor	Cross Validated EER (%)	Dimensions	EER Improvement (%)
Optimal Geometric	2.21111	20	-65.8
Best Holistic	2.71111	15	-6
Best Spectral	3.4	14	-68.1
Best Wavelet	1.7	75	-31.8

Table 5.3: This table demonstrates the change in performance achieved by the Z-Score normalizer against the best performing feature extractors from chapter 4.

Looking back at tables 5.1 through 5.3, we demonstrated that by performing normalization we could increase the GRF recognition performance in three of our feature extractors. Yet our results varied greatly and only the LTN normalizer was able to achieve improved performance for each of its applicable feature extractors. It should be noted that these normalizers modeled our dataset in a way that allowed each sample to be compared in a common scale, but none of them were capable of actually modeling the suggested relationship between the step duration and the shape of the 8 GRF signal curves. In [6] it was suggested that such a relationship exists, and, if this were the case, then by using a normalizer capable of learning and modeling this relationship we may be able to significantly improve the GRF recognition performance for our chosen feature extractors. In the next section we examine this relationship and its application in GRF normalization.

5.2 Regression

To discover and apply the proposed relationship between step duration and the GRF signal curves, we developed a new normalization approach based on the derivation of regression models. In this new approach, which we refer to as Localized Least Squares Regression (LLSR), we derived a set of models able to predict the position that each feature in the dataset would be expected to occupy were the underlying sample step durations aligned. The ability to convincingly and consistently demonstrate an increase in GRF recognition performance using this technique would, together with the results of our LTN in the previous section, support one of the two primary assertions of this thesis: that a relationship, useful to recognition, exists between the step duration and the GRF force signature.

The LLSR normalization technique draws from an area of data analysis known as analysis of covariance (ANCOVA) [60]. Using ANCOVA, the effect of the covariate, a variable that has a predictable influence on a data sample being analyzed, is removed from a set of analyzed samples. This is accomplished by aligning each sample according to the linear relationships that model the location of sample features with respect to the examined covariate. In our GRF dataset we treat the step duration, the total time recorded from the instant the heel first touches the force plate to the time the toe exits the force plate, as the covariate. In the ideal scenario, as shown in figure 5.2, if we knew of a perfect linear relationship between the step duration and the amplitude of a GRF feature, then removing the differences in step duration between the set of samples for any given subjects would result in the complete removal of intra-subject variance leaving only the

inter-subject variance remaining. However, in a practical scenario, given our GRF data, the covariate alignment would simply be expected to result in a proportionate decrease of intra-subject variance with respect to inter-subject variance.

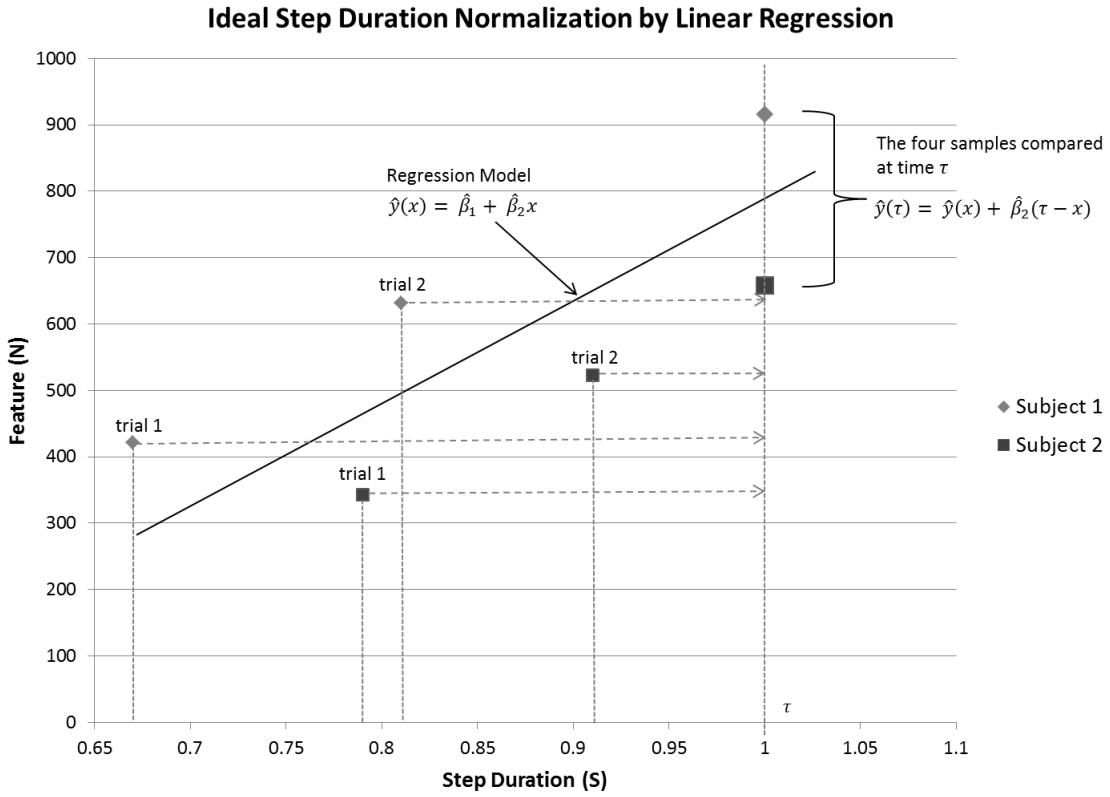

Figure 5.2: This figure demonstrates the ideal application of a linear model to normalize the features for four samples with respect to step duration.

To derive the proposed relationship between the step duration covariate and our data samples, shown in equation 5.5, we use a regression estimation approach known as Least Squares regression (equation 5.6). This technique obtains an estimate of the linear relationship between two or more variables by solving for the vector ($\boldsymbol{\beta}$) that minimizes sum of the residuals (or shortest distances) between itself and the locations of the

examined features. In our LLSR normalizer, we solve for the linear relationship between each feature and the step duration using pooled within group regression [60]; this approach treats differences in slope (β_2) between subjects as insignificant and pools the samples from all subjects into the single group for regression analysis. Although pooled within group regression can lead to the loss of subject specific information and, ideally, we would have calculated individual regression slopes for each subject, in a typical biometric scenario we would not have enough training samples per subject to generate statistically meaningful results from a subject specific approach.

$$y_{ij} = f(t_{ij}) - f(\min_{k \in N_i}(t_{ik})) \qquad x_{ij} = t_{ij} - \min_{k \in N_i}(t_{ik}),$$

$$\boldsymbol{y} = \boldsymbol{X\beta}, \quad \text{where} \quad \boldsymbol{y} = \begin{bmatrix} y_{11} \\ y_{12} \\ \vdots \\ y_{mn} \end{bmatrix}, \quad \boldsymbol{X} = \begin{bmatrix} 1 & x_{11} \\ 1 & x_{12} \\ \vdots & \vdots \\ 1 & x_{mn} \end{bmatrix}, \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix}$$

Equation 5.5: This equation demonstrates a linear estimate of the relationship between total step duration t_{ij} and the amplitude of a given feature $f(t_{ij})$ for a given subject i and trial sample j, where N_i represents the set of trial samples for subject i. The linear estimate of the relationship between y and X in our development dataset is modeled by $\boldsymbol{y} = \boldsymbol{X\beta}$, where $\boldsymbol{\beta_1}$represents the y-intercept and $\boldsymbol{\beta_2}$the linear slope.

$$\widehat{\boldsymbol{\beta}} = (\boldsymbol{X^T X})^{-1} \boldsymbol{X^T y}$$

Equation 5.6: To solve for the least squares approximation of the intercept and slope (represented in the vector $\widehat{\boldsymbol{\beta}}$) the linear representation from equation 5.5 can be rearranged as shown above.

Prior to performing our regression, to reduce the potential for undesirable data representations being introduced due to the inclusion of multiple subjects in our

regression analysis, we adjust the sample features such that the sample with the shortest step duration for each subject has its features shifted to the coordinate (0, 0). Next, we use the same shifting transform to shift all features in each of the other samples belonging to respective subject so they end up in the same relative position with respect to the zero-centered sample; this can be thought of as a type of calibration (see figure 5.3). Finally, after we have calculated the regression vector for a feature and its covariate, we can then 'remove' the influence of step duration on that feature by normalizing all samples containing the feature to the training dataset's mean step duration, as per equation 5.7.

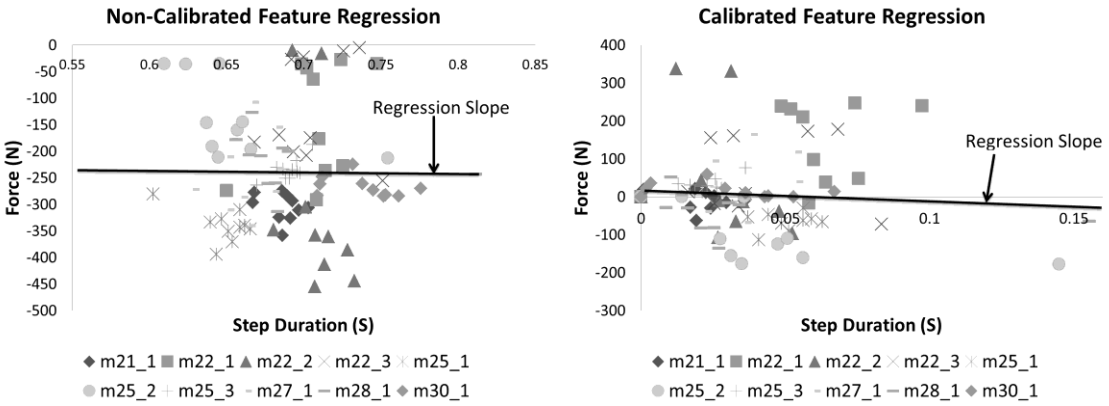

Figure 5.3: This figure demonstrates how a relationship between step duration and feature force becomes more apparent after "calibrating" the data so that the shortest step duration for each subject is placed at point (0,0) and all other feature samples for that same subject are scaled around it. In this case we find the force amplitude for the given feature has a tendency to decrease as step duration increases.

$$\hat{y}(x) = \hat{\beta}_1 + \hat{\beta}_2 x\,, \qquad \hat{y}(\mu) = \hat{\beta}_1 + \hat{\beta}_2 \mu$$

$$\hat{y}(\mu) - \hat{y}(x) = \hat{\beta}_1 + \hat{\beta}_2 \mu - (\hat{\beta}_1 + \hat{\beta}_2 x)$$

$$\hat{y}(\mu) = \hat{y}(x) + \hat{\beta}_2 (\mu - x)$$

Equation 5.7: Given the relationship derived in the previous two equations, we can model the expected location of a feature $\hat{y}(x)$ for a given step duration *x* using the *y*-intercept $\hat{\beta}_1$ and slope $\hat{\beta}_2$. If we know the location of a feature and step duration for the sample the feature was recorded in, then, for the dataset's mean step duration μ, we can find the expected position of its respective feature using the above relationship.

Having demonstrated how regression can be used to normalize an individual feature, our LLSR normalizer can be described as the application of this process to every feature across our entire dataset feature space. We first calculate the set of regression slopes modeling the relationship between the amplitude of each individual feature and step duration. Then we use the discovered slopes together with our feature position modeling function in equation 5.7 to derive our amplitude warping function (equation 5.8). Finally, step duration-based normalization can be accomplished by simply passing our samples directly into the amplitude warping function.

$$S_i = \{s_{i1},\ s_{i2},\ s_{i3} \ldots s_{iN}\}\,, \qquad s_{ik} \approx A_{1k} + A_{2k} t_i, \qquad S_i \epsilon\, S$$

$$\psi(S_i) = \left\{s_{i1} + A_{21}(t_\mu - t_i),\ s_{i2} + A_{22}(t_\mu - t_i)\ \ldots\ s_{iN} + A_{2N}(t_\mu - t_i)\right\}$$

Equation 5.8: This equation demonstrates the step duration-based amplitude warping function (ψ) for an *N*-dimensional sample *i* with a total step duration t_i and amplitude regression slope A_{2k} for feature k. The sample belongs to a sample set *S* with a mean step duration t_μ.

For LLSR to be effective each sample in the dataset must have the same number of features and these features must be roughly proportional with regards to their positions in phase of the GRF curve. When features are not aligned according to phase the regression calculation and amplitude warping function will reflect undesirable phase information. Our geometric features were heuristically selected such that the geometric features for each sample lined up according to phase; but, in the case of our holistic, spectral and wavelet feature spaces, only the area-based resampling approaches provided an approximate alignment across samples (samples were roughly aligned by resampling to a set of areas prior to extraction). However, this limitation did not pose a problem when assessing the performance of our LLSR normalizer because our best performing feature spaces all used the area-based resampling approach.

The results of running the LLSR normalization with our best performing feature extractors are demonstrated below in table 5.5. Examining these results we can see that the application of the LLSR normalizer to the optimization of the geometric feature set produced an 85.1% reduction in the EER when compared with the non-normalized optimal geometric feature extraction results. Figure 5.4 demonstrates a visualization of a subset of this new feature space formed with best 3 LLSR-normalized optimal geometric features, while table 5.4 demonstrates the complete list of optimal geometric features that make up the space. By comparison, each of our non-geometric feature extractors performed poorly when used in conjunction with the LLSR normalizer. It was previously noted that the alignment of features with respect to sample phase was implicit in the geometric feature extractor but approximated by a form of resampling in our other feature

extractors. If we consider the shape of the GRF phase to be something that can vary between different subjects and even between samples of the same subject (when taken at different walking speeds) then our LLSR regression normalization technique will end up generating its step duration-to-feature amplitude warping function models using a distorted feature alignment, even if all samples were resampled to the same number dimensions. To address this potential source of error, in the next section we present an extension of the LLSR normalizer which performs an additional alignment step prior to the generation and application of the regression models.

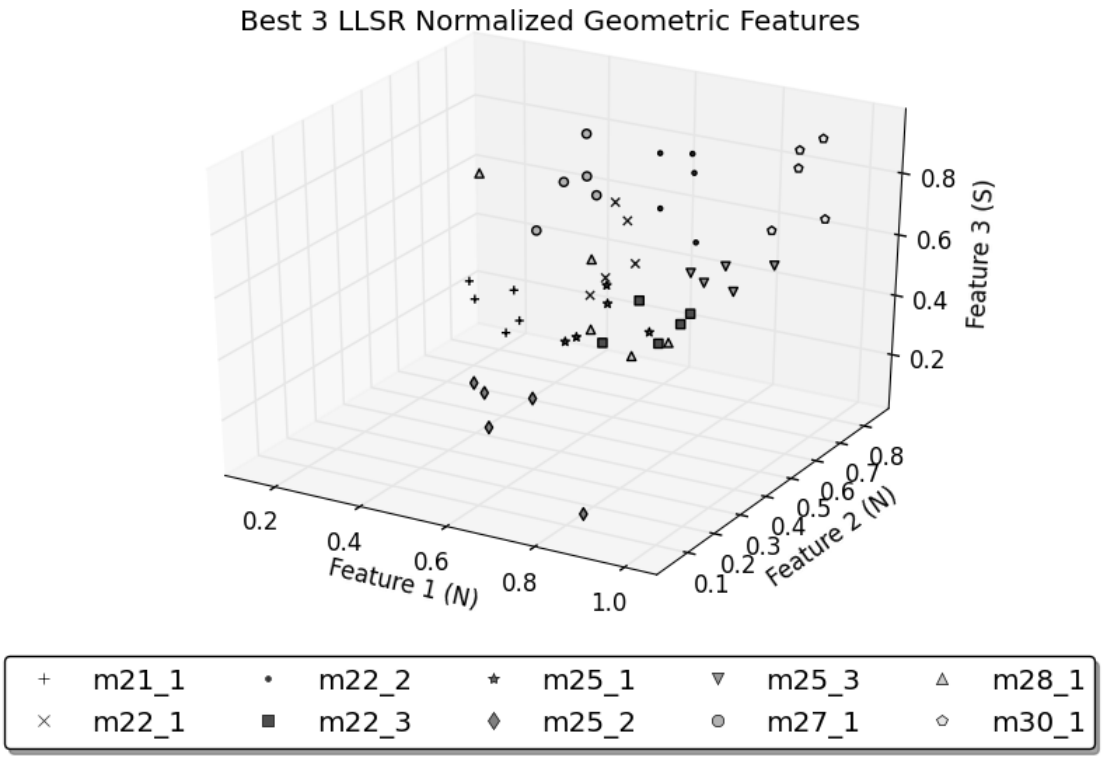

Figure 5.4: This figure demonstrates the three features in the geometric feature set that, when combined, result in the best GRF recognition performance. For visualization purposes we have scaled each feature to fall within the range of 0 and 1. When compared with our non-normalized optimal geometric features these three features demonstrated a 39% increase in recognition performance.

LLSR Normalized Optimal Geometric Features

Feature	Unit	Feature	Unit
$D_{F1Y2MIN2\_F1X1MAX1}$	FORCE	$D_{F1Z3MIN1\_F1Z3MAX2}$	FORCE
$D_{F1Y1MIN2\_F1Y2MAX1}$	FORCE	$D_{F1Y1MAX1\_F1Y2MIN2}$	TIME
$D_{F1Z1MAX2\_F1Y2MIN2}$	TIME	$D_{F1Z3MIN1\_F1Y1MIN2}$	TIME
F1Y1 MIN1	FORCE	$D_{F1Z1MAX2\_F1Z2MAX1}$	TIME
$D_{F1Z2MAX1\_F1Z2MAX2}$	FORCE	$D_{F1Y1MIN1\_F1X1MAX1}$	TIME
$D_{F1X2MAX1\_F1X2MIN1}$	FORCE	$D_{F1Z2MAX1\_F1X1MAX1}$	TIME
$D_{F1Z2MAX1\_F1Z4MAX2}$	TIME	$D_{F1Z4MAX1\_F1Z4MIN1}$	TIME
$D_{F1Y2MIN2\_F1X2MIN1}$	FORCE	$D_{F1Z1MAX2\_F1Y1MIN1}$	TIME
F1Y1 MIN2	TIME	F1Y2 NORM	FORCE
$D_{F1Z1MAX1\_F1Z1MAX2}$	FORCE	$D_{F1Y2MIN1\_F1X1MIN1}$	FORCE
F1Y2 MIN2	FORCE	F1X1 NORM	FORCE
$D_{F1Y1MAX1\_F1X1MAX1}$	FORCE	$D_{F1Z1MIN1\_F1Y2MAX1}$	TIME
$D_{F1Z1MIN1\_F1Z3MAX2}$	TIME	$D_{F1Z1MAX2\_F1Y1MIN2}$	TIME
$D_{F1Z4MIN1\_F1X1MAX1}$	TIME	$D_{F1Z1MAX1\_F1Y2MIN2}$	TIME
$D_{F1Z1MAX2\_F1X1MAX1}$	TIME	$D_{F1Z2MAX1\_F1Z3MAX2}$	TIME
$D_{F1Y1MAX1\_F1Y2MIN2}$	FORCE	$D_{F1Z1MAX1\_F1Y1MIN2}$	TIME
F1Y2 MIN2	FORCE	$D_{F1Y2MIN2\_F1X2MAX1}$	FORCE
$D_{F1Z1MAX1\_F1Y1MAX1}$	TIME	$D_{F1Y1MAX1\_F1X1MAX1}$	TIME
$D_{F1Y1MIN1\_F1X1MIN1}$	FORCE	$D_{F1Y2MIN1\_F1Y2MIN2}$	FORCE
$D_{F1Z3MAX2\_F1X1MAX1}$	TIME	$D_{F1Z1MAX2\_F1Z3MAX1}$	TIME
$D_{F1Y2MIN2\_F1X1MAX1}$	TIME	$D_{F1Z1MAX2\_F1Z3MAX2}$	TIME
$D_{F1Z2MAX1\_F1Y2MIN1}$	TIME	$D_{F1Z3MAX1\_F1Z3MIN1}$	FORCE
$D_{F1Z1MAX1\_F1Z3MAX2}$	TIME	$D_{F1Z2MAX1\_F1Z4MAX1}$	TIME
$D_{F1Z3MAX2\_F1Z4MAX2}$	FORCE	$D_{F1Z2MAX1\_F1Y1MIN1}$	TIME
$D_{F1Z4MAX1\_F1Z4MAX2}$	TIME	F1X2 NORM	FORCE
$D_{F1Z1MAX2\_F1Z4MAX2}$	TIME	$D_{F1Y1MIN2\_F1X1MAX1}$	FORCE
$D_{F1Z1MAX2\_F1X2MAX1}$	FORCE	F1Z1 MAX2	TIME
$D_{F1Y2MIN1\_F1Y2MAX1}$	FORCE		

Table 5.4: This table demonstrates the geometric features that were determined to be optimal for GRF recognition using the notation presented in the geometric feature extraction section of chapter 4.

LLSR Normalized Feature Extractors

Feature Extractor	Cross Validated EER (%)	Dimensions	EER Improvement (%)
Optimal Geometric	0.17777	55	86.6
Best Holistic	2.53333	15	0.8
Best Spectral	2.6	16	-28.5
Best Wavelet	2.33333	87	-81

Table 5.5: This table demonstrates the change in performance achieved by the LLSR normalizer against the best performing feature extractors from chapter 4.

5.3 Dynamic Time Warping

In the previous section we found that the LLSR normalization technique was less effective on our non-geometric feature spaces than it was on our geometric feature space. To improve upon the LLSR-normalized GRF recognition performance in our non-geometric feature spaces we adapted the technique to perform a non-linear sample alignment prior to the generation and application of our regression models. This new technique, which we refer to as Localized Least Squares Regression with Dynamic Time Warping (LLSRDTW), uses the two-sample Dynamic Time Warping (DTW) alignment technique together with the multi-sample center star alignment algorithm to accomplish the desired regression alignments. The LLSRDTW normalizer generates two sets of regression models, one reflecting the effect of step duration on the GRF curve amplitude and the other reflecting its effect on the GRF curve phase. These regression models are generated using a center star aligned training dataset, and, during sample recognition, tested samples are mapped against the center star-aligned training dataset using DTW. The resulting mapping is in turn used to produce phase and amplitude warping functions for the given samples, and the LLSRDTW normalization process is completed by first transforming the samples using the amplitude warping function then passing the amplitude-warped sample through the phase warping function to produce our normalized sample.

To understand the LLSRDTW it is important that we first understand the DTW algorithm that sits at the core of both its warping functions and center star alignment algorithm. The DTW algorithm used in this thesis takes two samples and derives the non-linear scaling

(or path) for each that minimizes the distance between the two. To do this, we first map every feature in one of the samples to every feature in the other and mark each of the mapped feature pairs with the cost generated by the pair's global cost function (equation 5.10). After mapping an *n*-dimensional sample to an *m*-dimensional sample we would get an $n \times m$ grid like the one shown in figure 5.6. In general, a low cost implies that there exists a path to the given pair with strong sample phase alignment. To calculate the optimal DTW path there are four constraints that we take into consideration: the path cannot go backwards in time (to smaller feature indices), each feature index must be included in the path at least once, for any given feature position in the DTW path the next position must come from an immediate neighbouring feature index pair, and the cost of a feature pair must reflect its local cost as well as its global cost.

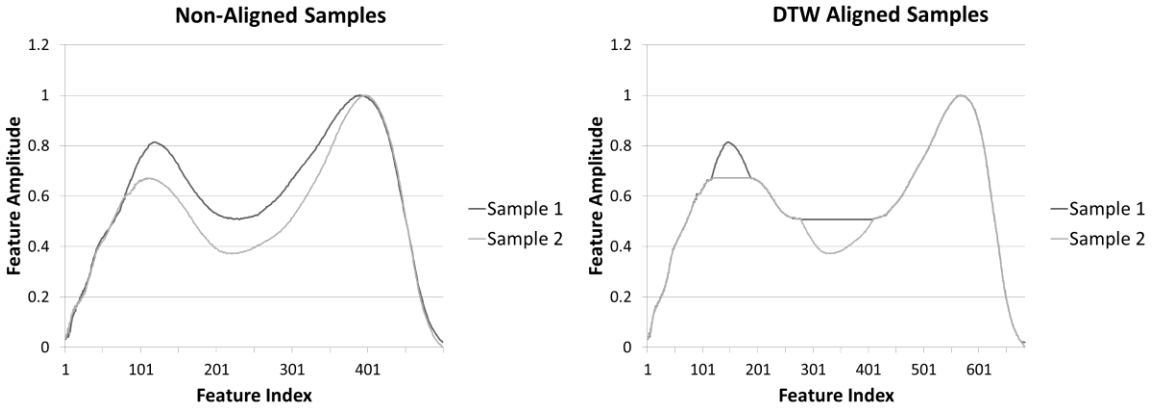

Figure 5.5: These graphs compare the non-aligned signals from figure 5.6 to those aligned using the derived DTW path. Although the alignment is rigid, it is guaranteed to be minimum cost.

$$d(i,j) = \left|\frac{f_1(i)}{\max(|f_1(x)|)} - \frac{f_2(j)}{\max(|f_2(x)|)}\right|$$

Equation 5.9: This equation demonstrates the local DTW cost function for the pair of values at index *i* of sample 1 (*f*1*(x)*) and index *j* of sample 2 (*f_2(x)*), where sample 1 and sample 2 are to be aligned via DTW.

$$D(i,j) = d(i,j) + \min\begin{Bmatrix} D(i-1,j-1) \\ D(i-1,j) \\ D(i,j-1) \end{Bmatrix}$$

Equation 5.10: This equation demonstrates the global DTW cost function for the pair of values at index *i/j* of samples 1 and 2, respectively. The cost function is initialized at *i, j=0*, and the value returned for a given *i/j* reflects the cost of the best alignment to the pair.

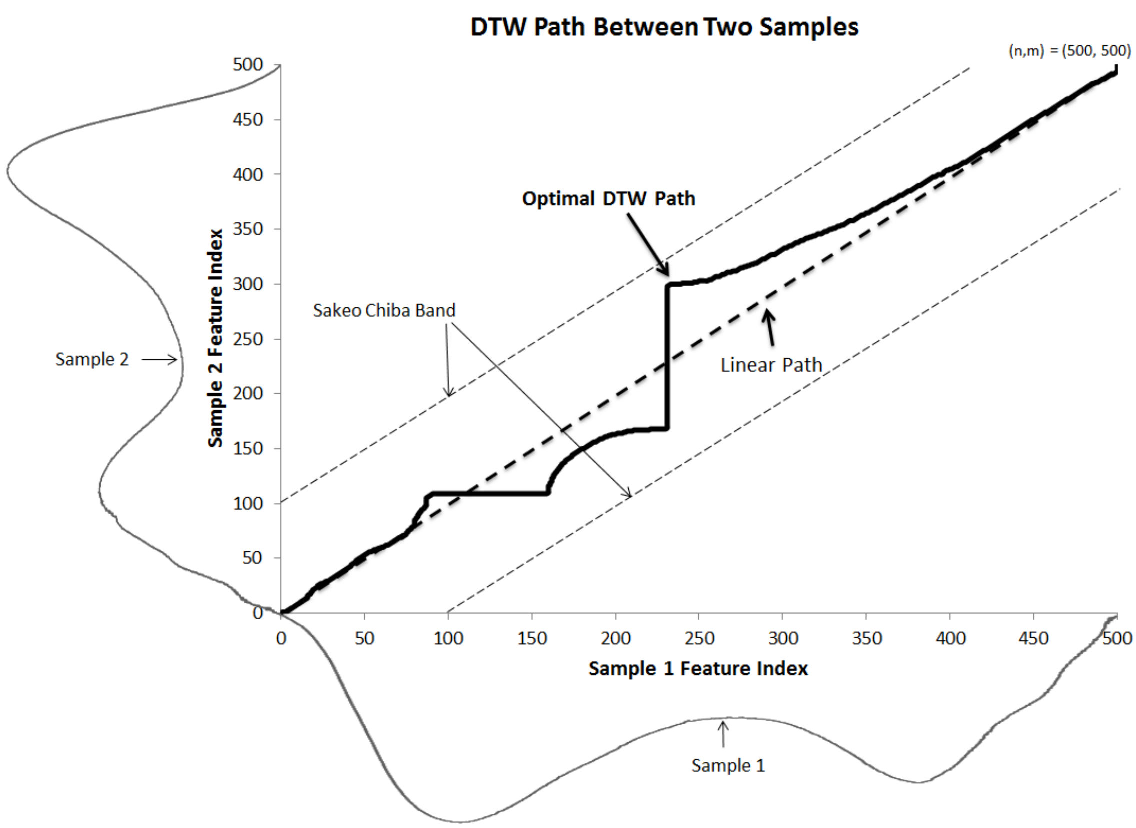

Figure 5.6: This chart demonstrates how the indices of two samples can form a grid of index pairs. It also demonstrates the optimal path between the two samples, discovered after calculating the global costs for each index pair. This chart also demonstrates the Sakeo

Chiba Band, which can be used to constrain the allowable deviation of the optimal path from the linear path.

In figure 5.6 we demonstrated an optimal warping path between an *n*-dimensional and *m*-dimensional sample. This path was derived by stepping through the index pairs with the minimum global cost from position (0,0) to position (*n*,*m*). The resulting DTW path derivation accounts for differences in scale and/or sample length by repeating the path indices for one of the samples in regions where its phase becomes misaligned with the other. When the process is finished, the set of DTW feature pairs can be split into a new sample space, with both samples approximately aligned according to phase, as shown in the graph on the right in figure 5.5. To guide the alignment of our sample GRF signals, we used the absolute difference function as our local cost function (equation 5.9), and, to improve both the speed and alignment performance of DTW, we used a global path constraint known as the Sakoe-Chiba Band [61]. The Sakoe-Chiba Band marks the maximum degree to which the DTW path can deviate from a linear path between the two samples, thus reducing the number of costs that need to be calculated and the potential for warping paths that are too distorted to be useful. As a final point, using a technique suggested in a study of DTW for speech recognition by Wang and Gasser [62], our implementation also decreased the potential for path distortions that could arise due to the variation of the sample amplitude by applying the L^{∞} norm (sup-norm) during the cost calculation. Consequently, our DTW path is calculated as though the samples used to generate it were aligned according to maximum amplitude when, in fact, the algorithm's output samples do not have their amplitude scaled to any degree.

To this point we have shown how DTW algorithm can be used to align any two samples, but to generate our LLSRDTW regression models we required the alignment of multiple samples. Multi-sample alignment can be accomplished with a form of generalized DTW [63]; however, such an alignment would incur an exponential time complexity penalty making it infeasible for our purpose. Instead, for the purpose of our research, we implemented a polynomial time approximation of generalized DTW known as the center star algorithm [64]. Using the center star algorithm, if we were given a sample set S of k samples, $S_1, S_2, \ldots S_k$, the first step in the algorithm would be to determine the pairwise alignment costs for every possible combination of samples. In our case these costs are found by calculating the DTW cost grid for every possible sample pair and identifying the pairs with the DTW global cost found at the end point of each alignment. A tabulation of these costs shown in figure 5.7 demonstrates how, after calculating these pairwise costs, we can expose the single sample S_j with the minimum cost to all others. For the second step of the algorithm, we create a new aligned sample set S' by first adding our minimum cost sample to position S_1'. The third and final step of the center star algorithm then involves iteratively aligning and including the remaining samples in the aligned space. For each iteration i from 1 to k, where $i \neq j$, we calculate the DTW alignment $DTW(S_1', S_i)$ to get the two aligned samples S_1'' and S_i'; we then repeat the indices of all samples $S_{r>1}'$ currently in S' at any position for which a S_1' index was repeated in the generation of S_1''; finally, we set S_1' to S_1'', add S_i to S', and repeat the process for $i \leq k$. The set S' that results from completing this process represents the center star algorithms approximate alignment of S and we refer to the final value for S_1' as the aligned set's center star template.

	S_1	S_2	S_3	S_4	S_5
S_1	0	4.8226	9.2145	25.791	14.411
S_2		0	17.131	15.362	9.6685
S_3			0	44.824	32.025
S_4				0	4.9474
S_5					0

$\sum_{i=1\ldots5} D(S_1,\ S_i) = 54.23973$
$\sum_{\boldsymbol{i=1\ldots5}} \boldsymbol{D(S_2,\ S_i)} = \mathbf{51.37646}$
$\sum_{i=1\ldots5} D(S_3,\ S_i) = 118.003$
$\sum_{i=1\ldots5} D(S_4,\ S_i) = 90.92532$
$\sum_{i=1\ldots5} D(S_5,\ S_i) = 61.05241$

Figure 5.7: This figure demonstrates the tabulation of global costs between a set of example samples (S_1 to S_5). The sum of the costs from any given sample to all other samples in the sample set is demonstrated to the right of the table. In this case we can see that the sample S_2 has the minimum cost in relation to all others.

As previously mentioned, the LLSRDTW normalizer uses the center star approximation to align our training dataset prior to the generation of our step duration-to-phase and step duration-to-amplitude regression models. While this alignment could potentially be done across the entire GRF sample feature space, we instead determined it would be more effective to apply the center star approximation toward the alignment of each of our 8 GRF signal subspaces individually. The generation of our regression models over the resulting aligned training sample space produces 16 phase/amplitude regression models and 8 center star templates, all of which the LLSRDTW normalizer retains to use later in the derivation of its phase and amplitude warping functions. An example of the center star alignment for three of our GRF signals is shown in figure 5.8, while the corresponding regression models and center star templates are shown in figure 5.9. At this point we must revisit the step duration-to-phase regression models. Unlike our LLSR normalizer which generated its models under the assumption that the modeled sample space was already aligned according to phase, our LLSRDTW was able to determine the degree to which the phase of each sample was misaligned by comparing the phase

distortions required to align them. The original training sample set, which we pass to the center star algorithm, must be of the same format of that used by the LLSR normalizer, a dataset for which all samples have been resampled to a common length. The alignment will, by repetition of indices, alter the relative position of each original feature index with respect to the position of the same feature in the other samples. Consequently, we can use the center star aligned sample space to derive the Least Squares amplitude and phase regression vectors A^p and θ^p, which model the effect of step duration on both amplitude and phase, respectively, as demonstrated in equation 5.11. For computational efficiency, as a final step in our implementation, the regression models (A_2^P, θ_2^P) and center star templates are resampled to the dimensionality of the original dataset.

$$S_i^c = \{s_{i1},\ s_{i2} \dots s_{ik} \dots s_{iN}\}, \qquad k \epsilon\, CS(S_i), \qquad N = |S_i|, \qquad S_i^c \epsilon\, S^c$$

$$s_{ik} \approx A_{1K}^P + A_{2K}^P t_i, \qquad k \approx \theta_{1k}^P + \theta_{2k}^P t_i$$

Equation 5.11: This equation demonstrates the step duration-to-feature amplitude regression vector A^p and step duration-to-feature phase regression vector θ^p. In this equation A^p was derived by applying the least squares regression (equation 5.6) to the set of aligned feature values in the center star aligned feature set S^c. While θ^p was derived by applying the least squares regression to the set of aligned feature indices k in the center star aligned feature set. We identify the step duration for a given sample feature set i as S_i, with the assumption that each sample in the original sample space has already been resampled to N-dimensions.

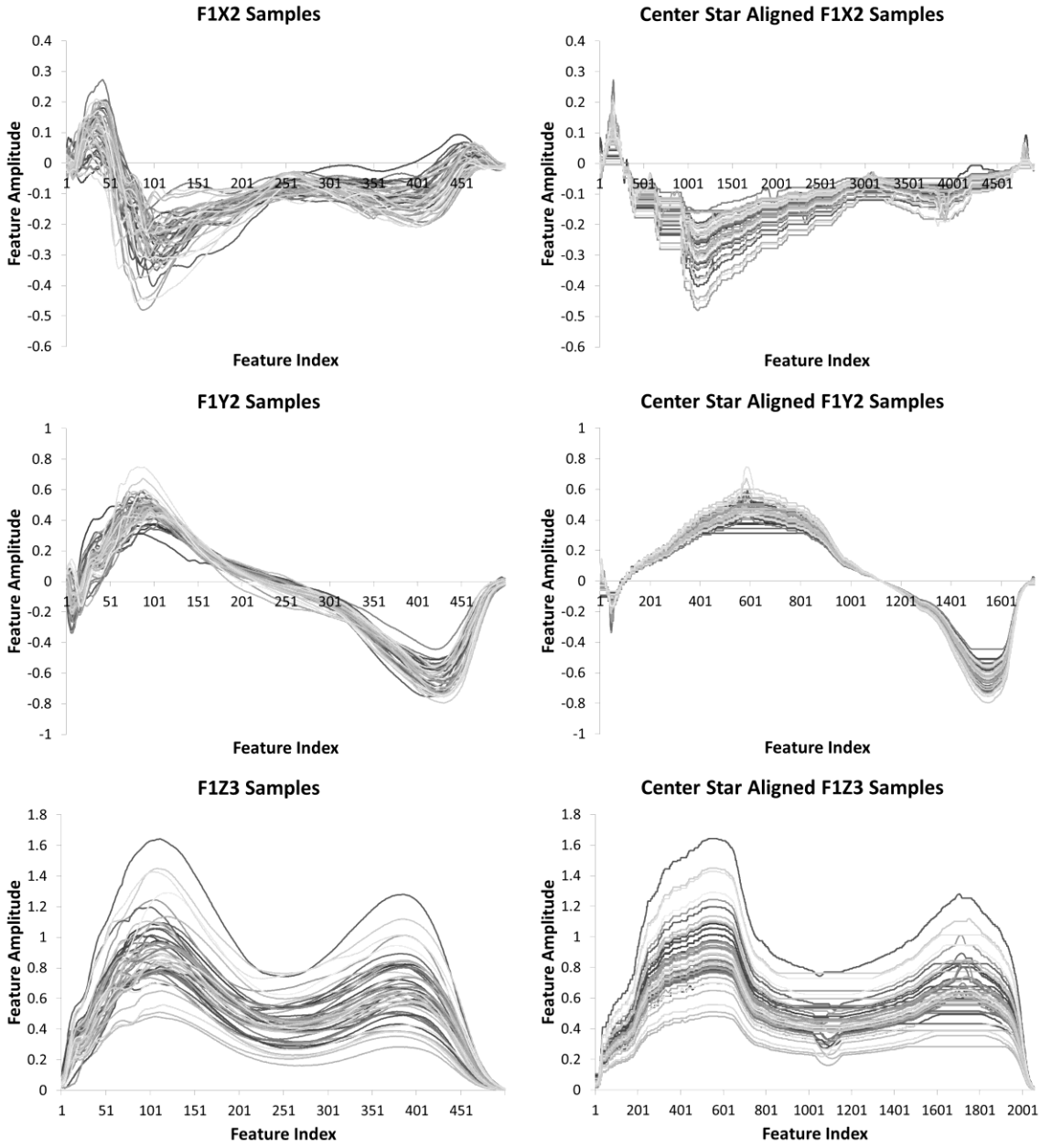

Figure 5.8: This figure graphically compares the center star aligned feature sets to the non-aligned feature sets for three different GRF signals in our 50 sample training dataset. The large expansion of indices that appear in the center star aligned feature sets are indicative of the repetition of feature indices required to perform the alignment.

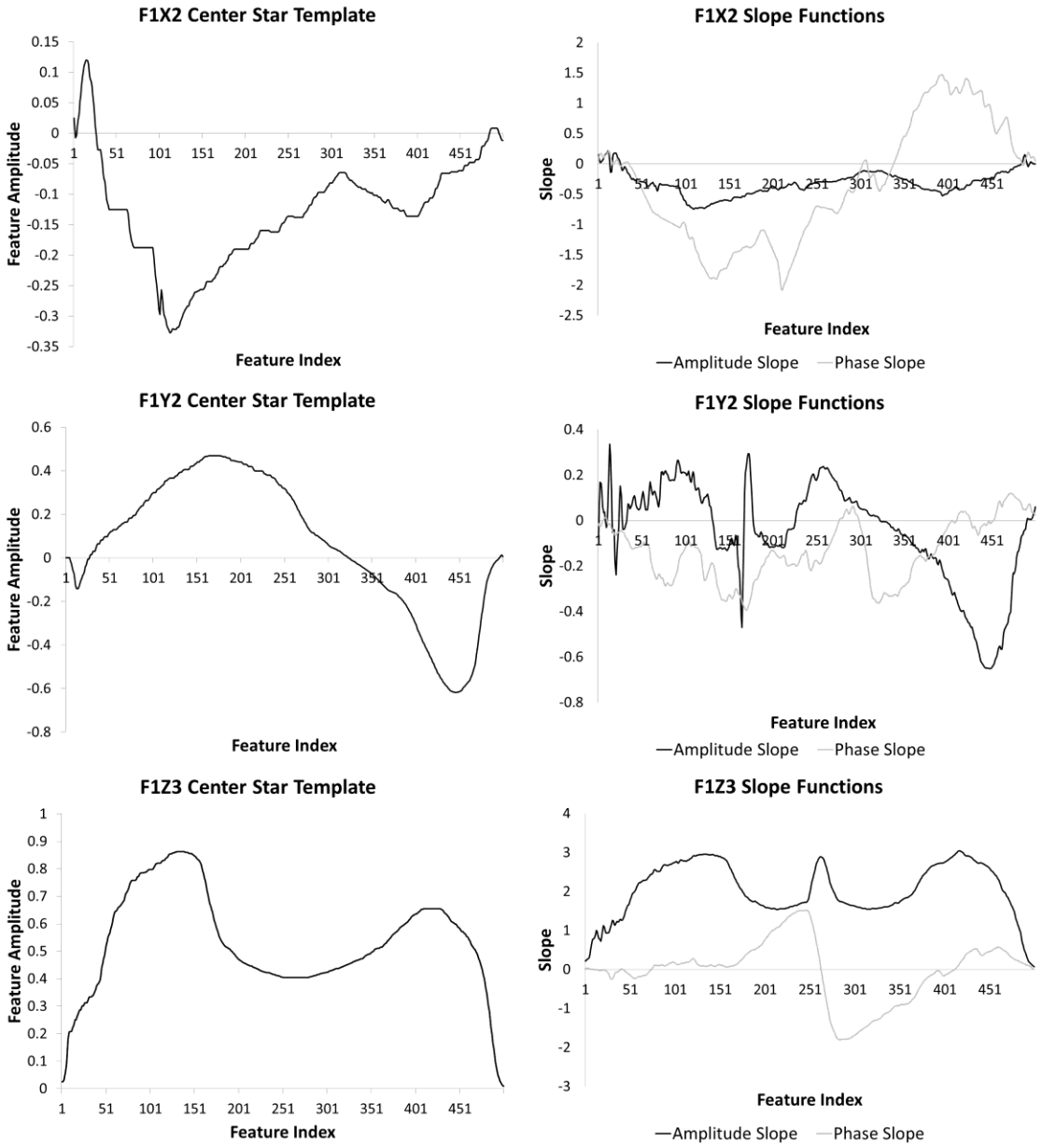

Figure 5.9: This figure demonstrates an example of the best center star template together with its corresponding amplitude and phase regression model (slope functions) corresponding to the 3 GRF signals alignments demonstrated in figure 5.8.

The LLSRDTW normalizer uses amplitude and phase warping functions to normalize samples with respect to step duration. But, since these functions depend on our regression models and our regression models reflect the center star-aligned feature space, we cannot

directly use them in the derivation of our warping functions like was done for the LLSR normalizer (equation 5.8). Instead, to ensure regression models are correctly aligned when included in our warping functions, we perform several additional preprocessing steps during the normalization of a sample. First, we find the DTW path between each signal in the sample and its respective center star template (equation 5.12). Next, we resample our regression models for each signal to the length of the aforementioned DTW paths. Finally, we use the DTW path mappings and resampled regression models together with an averaging technique to map the regression models into the original sample space (equation 5.13). Having mapped the regression models into the original sample space, the LLSRDTW amplitude warping function for any given signal in the sample will be almost identical to the simple warping function used by the LLSR normalizer (equation 5.14). The LLSRDTW phase warping function is slightly more complicated (equation 5.15). It first uses the resampled regression models to determine the degree to which the phase of the feature set must be warped to align it with the mean step duration, next it applies L^1 normalization to the result, and finally it shifts the L^1 normalized result such that the final result acquires a mean value of 1. These steps ensure the phase warping function only affects the shape of the curve but do not in any way alter its amplitude.

$$S_i = \{s_{i1},\ s_{i2},\ s_{i3} \dots s_{iN}\}, \quad S_i \epsilon\ S$$

$$S_i' = \{s_{i1}, \dots\ s_{ij}, \dots\ s_{iN}\}, \quad j \epsilon\ DTW_{S_c}(S_i), \quad M = |S_i'|$$

$$R = \{(1,1), \dots (j,m) \dots (N,M)\}$$

Equation 5.12: This equation demonstrates the application of DTW to align a feature set S_i with its respective center star template S_c. The resulting feature set S_i' represents feature set

formed by the DTW path and the relation *R* describes the mapping of indices between the *N*-dimensional original feature space and the *M*-dimensional DTW aligned feature space.

$$A_2^M = \{A_{21}^P, A_{22}^P, A_{23}^P \dots A_{2M}^P\}, \qquad \theta_2^M = \{\theta_{21}^P, \theta_{22}^P, \theta_{23}^P \dots \theta_{2M}^P\}$$

$$A_2^{N\prime} = \left\{\overline{A_{2R(1)}^M}, \overline{A_{2R(2)}^M}, \overline{A_{2R(3)}^M}, \dots \overline{A_{2R(N)}^M}\right\}, \qquad \theta_2^{N\prime} = \left\{\overline{\theta_{2R(1)}^M}, \overline{\theta_{2R(2)}^M}, \overline{\theta_{2R(3)}^M}, \dots \overline{\theta_{2R(N)}^M}\right\}$$

Equation 5.13: This equation demonstrates how the *P*-dimensional step duration-based amplitude and phase slope functions (A_2^P and θ_2^P) can be resampled to *M*-dimensions and then aligned to the *N*-dimensional feature set S_i from equation 5.12. In our implementation we resample the center star template, amplitude slope function and phase slope function to the dimensionality of the original dataset after generating our regression models (*P* = *N*).

$$\psi'(S_i) = \left\{s_{i1} + A_{21}^{N\prime}(t_\mu - t_i), s_{ik} + A_{22}^{N\prime}(t_\mu - t_i) \dots s_{iN} + A_{2N}^{N\prime}(t_\mu - t_i)\right\}$$

Equation 5.14: This equation demonstrates the step duration-based amplitude warping function (ψ') for feature set *i* with a total step duration t_i, belonging to the sample set *S* with a mean step duration t_μ. This amplitude warping function is derived by modifying equation 5.8 to use the *N*-dimensional amplitude regression slope derived in equation 5.13.

$$\delta(S_i) = \left\{\theta_{21}^{N\prime}(t_\mu - t_i),\ \theta_{22}^{N\prime}(t_\mu - t_i) \dots \theta_{2N}^{N\prime}(t_\mu - t_i)\right\}$$

$$\gamma(S_i) = \frac{\delta(S_i)}{\|\delta(S_i)\|_1}$$

$$\phi(S_i) = S_i \times \left(\gamma(S_i) - \overline{\gamma(S_i)} + 1\right)$$

Equation 5.15: The equation demonstrates the derivation of the phase warping function (ϕ) for feature set *i* with a total step duration of t_i, belonging to the sample set *S* with a mean step duration t_μ. The phase warping function should always be applied after the amplitude warping function, thus in our application our LLSRDTW normalizer can be described as the application of $\phi(\psi(S_i))$.

In our implementation, LLSRDTW normalization is performed on a given sample by first passing each of the sample's GRF signals through the amplitude warping function, then passing the resulting amplitude-warped feature sets into the phase warping function. To improve the speed and performance of our LLSRDTW normalizer we tested its underlying DTW algorithm with a set of Sakeo-Chiba Band values from one to twenty percent of the DTW cost grid size and discovered optimal bandwidths of 5%, 1%, and 10% for our best holistic, spectral and wavelet feature extractors, respectively. The results from the application of our LLSRDTW to our best non-geometric feature sets are demonstrated in table 5.6. Analyzing these results, we discovered that by better aligning our features we were able to achieve a modest increase in performance in all our feature extractors over the non-aligned LLSR normalization technique. However, only the LLSRDTW-normalized holistic and spectral feature spaces produced an increase in recognition performance over their non-normalized equivalents. The most notable outcome from performing LLSRDTW normalization was the performance increase achieved when it was applied to the spectral feature space. This technique achieved an increase in recognition performance where all other normalization techniques failed to increase performance. Consequently, our results suggest that by aligning each sample as though they were all taken with the same step duration we can improve GRF recognition performance in both heuristic and machine learning-based feature extractors, thus demonstrating the utility of the relationship between the step duration and the shape of the GRF curve.

LLSRDTW Normalized Feature Extractors

Feature Extractor	Cross Validated EER (%)	Band (%)	Dimensions	EER Improvement (%)
Best Holistic	2.3	5	13	9.9
Best Spectral	1.84444	1	17	8.7
Best Wavelet	1.45555	10	90	-12.9

Table 5.6: This table demonstrates the change in performance achieved by the LLSRDTW normalizer against the best performing feature extractors from chapter 4.

5.4 Summary

This chapter demonstrated how normalization techniques could be used to improve the selection of features for footstep GRF-based person recognition. In this chapter we discussed how variation in intra-subject GRF sample scale could potentially lead to feature extractors missing important features on account of their inability to distinguish differences due to variations in scale from those due to distinctive inter-subject characteristics. To address this issue we re-examined our best feature extractors from the previous chapter but this time with various normalization techniques applied prior to feature extraction. The normalizers examined used various methods to transform our footstep samples to a common scale in terms of both step duration and amplitude, including well known uniform scaling and shifting operations and two new dynamic techniques developed for the purpose of this research.

The two new normalization techniques introduced in this chapter (LLSR and LLSRDTW) were created to test our problem statement assertion that a potentially useful relationship exists between stepping speed (or step duration) and the GRF force signature. The LLSR normalizer attempted to model this relationship via the derivation of a series of individual regression functions, while the LLSRDTW was designed to improve upon the LLSR in machine learning-based feature extractors by using the technique known as DTW to align key sample data points prior to deriving the regression models. The best results obtained after applying these and the scaling and shifting normalizers to our top feature spaces in the development dataset are shown in table 5.7 (again we used the KNN classifier to acquire these results). Although the LLSRDTW

normalizer was found to result in a clear improvement over the LLSR normalizer in all applicable feature spaces it only proved to be the most performant normalizer in the spectral feature space. On a whole normalization was shown to improve GRF recognition in all feature spaces, but with regards to the best normalizer our results were divided with no two feature spaces achieving their best results over a shared normalizer. However, it must be noted that we were able to improve the recognition performance over each feature space when using a normalizer that accounted for step duration, which would appear to support our assertion regarding the relationship between stepping speed and the GRF force signature.

Normalizer GRF Recognition Performance

Feature Space	Best Normalizer	EER (%)
Optimal Geometric	LLSR	0.17777
Best Holistic	L1	2.04444
Best Spectral	LLSRDTW	1.84444
Best Wavelet	LTN	1.1

Table 5.7: This table compares the best GRF recognition performance achieved across each feature space when used in combination with a normalization technique.

In this chapter and the one proceeding we examined two parts of the biometric system that could jointly be described as data preprocessing. The application of these techniques makes it easier to identify important characteristics but they do not have the ability to distinguish GRF subjects on their own. In the next chapter we explore the biometric system component that is responsible for learning the patterns within the preprocessed data and using them to perform footstep GRF-based recognition.

Chapter 6

Classification

The final step in our biometric system after preprocessing the data with normalization and feature extraction involves the application of a classifier to perform recognition over the transformed sample space. Although classification could theoretically be applied directly to the sample set with no prior preprocessing we opted against doing so because, in addition to exposing features to the classifier that might otherwise be missed by overfitting, the use of our preprocessors also reduced the dimensionality of the samples' feature space to a size that could be effectively handled by all popular classifiers. Without needing to take computational efficiency into account, the goal of the classifier then became to find the boundaries in the transformed sample space that best separate the feature spaces of our subjects. Internally, classifiers use a number of tricks to discover these boundaries; however, classifiers are also subject to initialization parameters that can be adjusted to improve recognition performance. Knowing this, our classification goal, for any given classifier, can be addressed by solving an optimization problem; namely, discovering the classification parameters that optimize our recognition performance.

In our discussion in Chapter 2 we described how classifiers can be categorized as being based on either generative or discriminative models. We could also further distinguish classifiers as being either instance-based (lazy learning-based) or eager learning-based [65]. In instance-based classifiers the training phase involves storing all training samples

to memory and all calculations needed for recognition are performed at the time of recognition. Conversely, in eager learning-based classifiers the training phase involves the derivation of a function that is able to transform any given subject/sample pair into a representation of a decision reflecting whether the given subject corresponds to the given sample. In the following sections we examine classifiers covering each of the two classifier models and learning strategies. For each classifier we discuss the parameter optimizations that were used to improve recognition performance, while, as a final assessment of our various GRF recognition strategies we cross compare the full set of classifiers with our best performing feature extraction and normalization strategies from chapters 4 and 5.

6.1 K Nearest Neighbour

The most common category of classifier applied by previous GRF recognition studies [5, 39, 31, 7, 32, 3] was the K Nearest Neighbour (KNN) [66] classification technique, the classification technique used in the two preceding chapters to assist with the optimization of our feature extractors and normalizers. The KNN classifier is an instance-based discriminative classification technique that uses the distances between a given sample and its *K* nearest training samples to determine the sample's identity. The decision is based on a voting scheme under which the subject corresponding to the majority of the *K* nearest neighbours is identified as the owner of the sample. This technique comes in many variants and can be altered to perform recognition rather than identification by simply taking the expected identity as a parameter then accepting or rejecting the recognition requests based on the discovery of a matching identity in the majority of the *K* nearest neighbours.

The KNN recognition algorithm in its most basic form is relatively straightforward. During its training phase the classifier simply stores all training samples to memory. Subsequently, when performing recognition on a tested sample the KNN classifier begins by calculating the Euclidean distance (equation 6.1) between the sample and every sample in the training set. The training samples are then ordered according to their distance from the tested sample and the samples with the *K* smallest distances are used to represent a voting set. Finally, the identity assigned to the sample would correspond with the owner of the majority of samples in the voting set, or, in the event of a draw, the owner of the samples in the voting set with the shortest distance to the tested sample. If

this identity matches the identity provided for recognition then the algorithm would accept it as a match, otherwise it would be rejected. A simple visualization of the KNN identification process for a 2-dimensional feature space is demonstrated in figure 6.1. Using this simple identification scheme, the KNN classifier can be observed as having a key advantage over non-instance-based classifiers in that it does not need to be retrained when new samples are added-to or existing samples removed-from its training set.

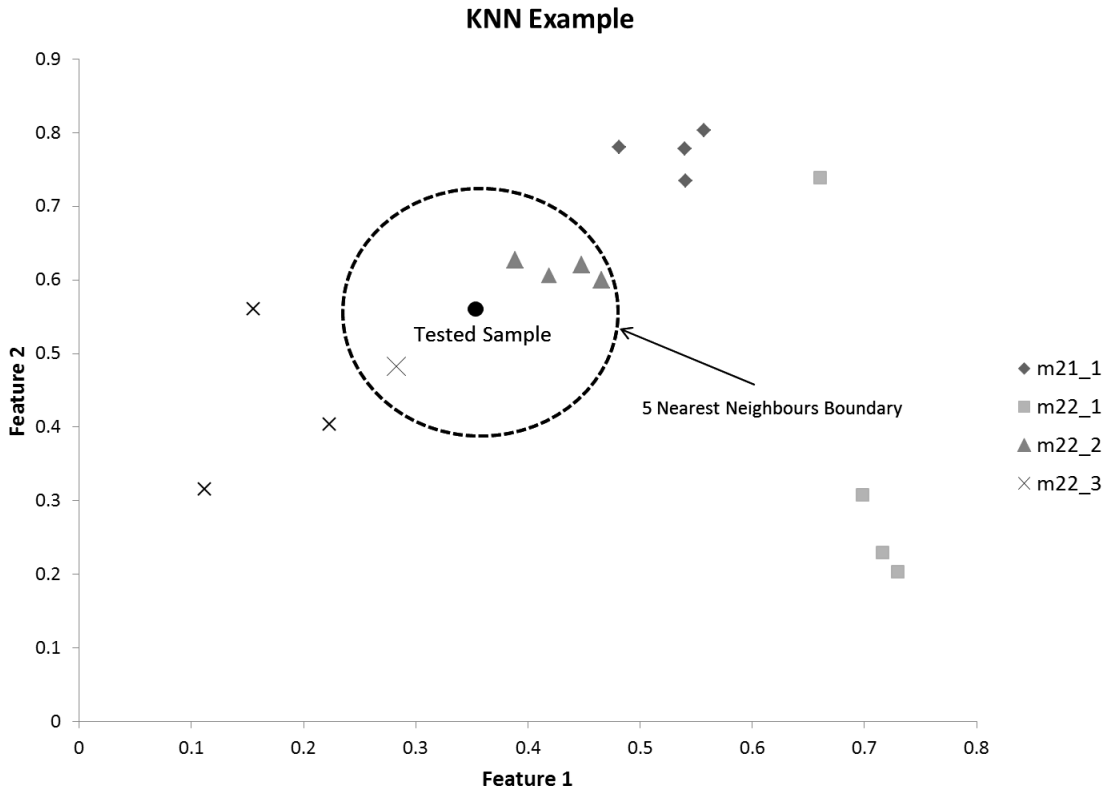

Figure 6.1: This graph demonstrates the samples that would form the voting set in a 2-dimensional feature space with the KNN *K* value set to 5. In this case only the subject m22_2 would be accepted as a recognition match for the given test sample.

$$d(\boldsymbol{p}, \boldsymbol{q}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

Equation 6.1: This equation demonstrates the Euclidean distance calculation used by our KNN classifier. In this equation the vectors *p* and *q* would represent the feature sets for two different samples.

In our experimental design we established that the verification EER would be used to assess GRF recognition performance, and that this value could be derived by first configuring our classifier to return posterior probabilities then scaling the classifier's acceptance threshold to the point at which the FAR and FRR intersect. The basic variant of the KNN classifier described in the previous paragraph does not return a probability, but, rather, a true or false decision based directly on the results of the voting scheme. However, in [32] Suutala and Röning demonstrated that posterior probabilities, which reflect the likelihood of a given subject matching a given sample, could be estimated by counting the occurrences of the given subject in the *K* nearest training samples for the given sample (equation 6.2). Unfortunately, the KNN posterior probability generation technique used in [32] can be subject to undesirable biases from outlier points when either the number of training samples per subject is limited or the value chosen for *K* is small. To address this problem we used a form of weighted KNN [66], for which the *K* samples in the voting scheme were given weights inversely proportional to their distances from the sample being recognized. Using this weighted KNN technique, we were then able to alter equation 6.2 to reflect the assigned weight values, as shown in equation 6.3, and thus acquire posterior probabilities less affected by small sample size-induced biases.

$$P(w_i|x) = \frac{p(x|w_i)p(w_i)}{\sum_{j=1}^{c} p(x|w_j)p(w_j)} \approx \frac{k_i}{k}$$

Equation 6.2: This equation demonstrates the posterior probability estimation based on occurrences for an unweighted KNN classifier. In this equation k_i represents the number of occurrences of subject *i* in the *k* nearest neighbours to tested sample *x* and $P(w_i|x)$ is the estimate of the probability density function.

$$P(w_i|x) \approx \frac{\sum_{j=1}^{k} \frac{\left(v_{k_j}=i\right)}{d\left(u_{k_j},x\right)}}{\sum_{j=1}^{k} \frac{1}{d\left(u_{k_j},x\right)}}, \qquad \left(v_{k_j} = i\right) = 1, \qquad \left(v_{k_j} \neq i\right) = 0$$

Equation 6.3: This equation demonstrates the posterior probability estimation based on distances for a weighted KNN classifier. In this equation the occurrences of subject *i* are weighted according to the distance between the occurrence u_{k_j} and the tested sample *x*. Only samples belonging to the tested subject *i* are counted in the numerator, as demonstrated through the values returned by the expression $v_{k_j} = i$.

There are two configurable inputs that must be accounted for when implementing the KNN classifier: the parameter *K* used to represent the size of the voting set, and the sample feature sets used in the training and testing of the algorithm. Examining our sample inputs we noticed that each of the features in the sample feature space was represented over a different scale reflecting the degree of absolute variance in the part of the feature space from which the feature was derived. Consequently, features represented in larger scales would gain a large influence on the classification result regardless of their discriminative ability. Ideally, each feature would be scaled in proportion to its discriminative ability. However, determining the correct proportions for such scaling is generally computationally infeasible, so instead we decided to use the approach taken in [32] and gave each feature equal influence in the recognition decision. We accomplished

this by finding the minimum and maximum values for each feature in our training dataset, setting these values to 0 and 1, and scaling all training and testing sample features with respect to these feature space boundaries. In addition to this input scaling we attempted to optimize the value of *K* to improve our recognition performance. To do so we used a brute force approach, comparing the cross-validated EER results returned for 10 different values of *K* across each of our best performing preprocessor configurations, as demonstrated in figure 6.2.

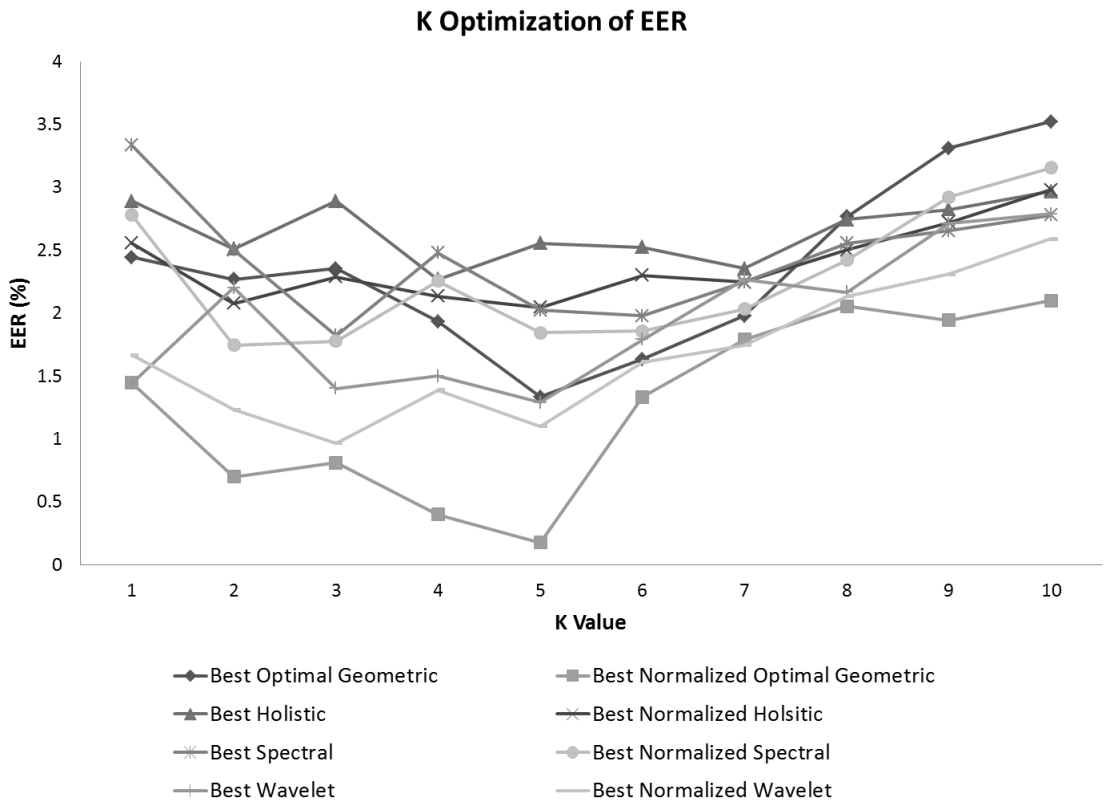

Figure 6.2: This figure demonstrates the cross-validated EER achieved by our best feature extractors and normalizers for the values of *K* from 1 to 10.

As we can see in figure 6.2, the KNN classifier tended to perform worse when the *K* value was greater than the number of samples used for training any given subject (5 in

our case). We also observed a tendency for our best results to occur when the *K* value was set to 5, though it must be noted that because our preprocessor optimization was done using a *K* value of 5 our results were positively biased toward this value of *K*. Yet, despite this bias we were still able to achieve better results in half of our preprocessors using other values of *K*, as shown in table 6.1. Comparing these findings with those in [3], the only previous GRF recognition study to identify an optimal *K* value, we found all but one of our *K* values represented a larger percentage of the number of samples used for training a single subject than that was found by Rodríguez et .al. Yet, the KNN classifier used in [3] was not weighted, and, of the previous GRF recognition studies, only [7] involved the use of a weighted KNN algorithm similar to the one used in this section (Fuzzy KNN [67]]). Thus our KNN-specific findings could not be directly compared with any previous GRF studies.

Optimal KNN Classifier Results

Pre-processor	K-Value	Threshold	Cross Validated EER (%)	EER Improvement (%)
Best Optimal Geometric	5	0.3359	1.33333	0
Best Normalized Optimal Geometric	5	0.3921	0.17777	0
Best Holistic	4	0.2765	2.26666	11.3
Best Normalized Holistic	5	0.3015	2.04444	0
Best Spectral	3	0.2515	1.82222	9.8
Best Normalized Spectral	2	0.2859	1.74444	5.4
Best Wavelet	5	0.1703	1.28888	0
Best Normalized Wavelet	3	0.2921	0.96666	12.1

Table 6.1: This table demonstrates the best performance achieved by the KNN classifier for each preprocessing technique. The threshold shown is the threshold at which the EER value was calculated (a value between 0 and 1 derived from the raw posterior probability output) and the EER improvement represents improvement in recognition performance achieved by the *K* value optimization over the results calculated in the previous two chapters.

6.2 Multilayer Perceptron Neural Network

In many classification problems the distributions of the classes being classified do not form the closely bundled symmetric clusters optimal for classification when using distance-based classifiers like the KNN. Instead the boundaries that separate the various classes (in our case subjects) may be irregular with sharp cut-off points as shown in figure 6.3. These boundaries can be difficult to define geometrically; however, they can usually be approximated to a high degree of accuracy using a machine learning structure known as an Artificial Neural Network (or Neural Network). Neural Networks derive from early attempts to model the problem solving abilities of the connected neurons in the human brain [68], and have proven particularly useful in the field of pattern recognition [69]. These networks are typically represented as a directed graph consisting of a set of interconnected processing units, or nodes (figure 6.4), with one subset of the nodes accepting input data and another output subset presenting the results of any processing that was performed as the input data passed through the network. The structure represented by the arrangement of these nodes can be categorized as being either feed-forward or recurrent. In feed-forward networks, information passes between the input nodes and the output nodes without ever passing through the same node twice, while in recurrent networks, processed output will end up getting passed back into nodes that have already seen the original input, allowing the network to become aware of state. For the purpose of our research, we based our GRF Neural Network analysis on a popular forward-feed network structure known as a Multiple Layer Perceptron (MLP) [70], a structure which previously achieved strong GRF recognition performance in [32].

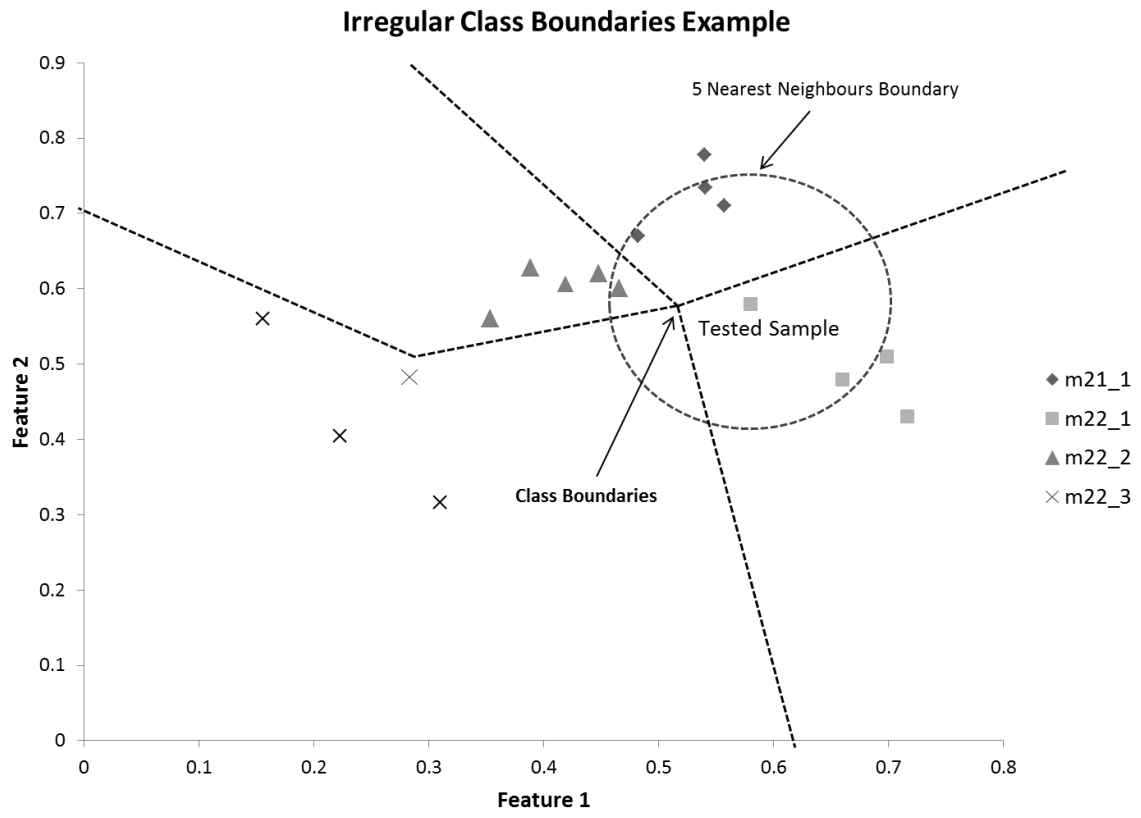

Figure 6.3: This figure demonstrates a sample "Tested Sample" that would be falsely attributed to the wrong class by the KNN classifier, but would be correctly matched were the identified class boundaries used during classification.

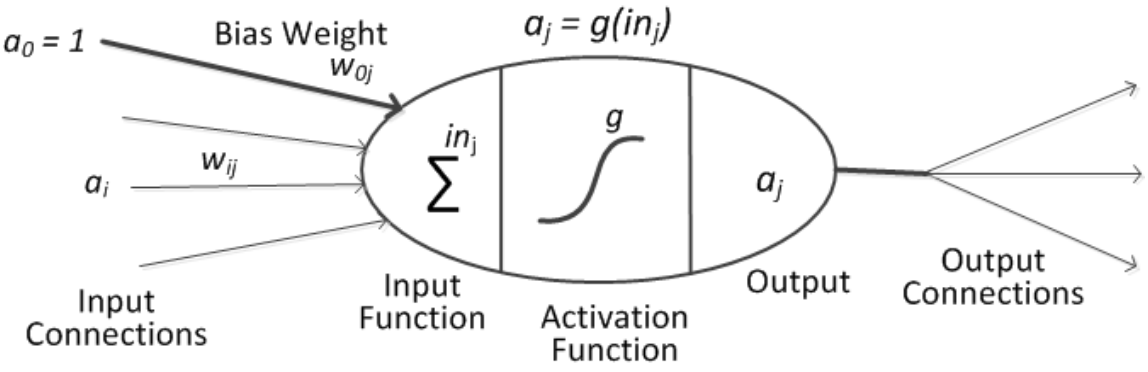

Figure 6.4: This figure demonstrates an individual ANN node. The node accepts a series of weighted inputs including a bias input, which increases or lowers the net input of the activation function where needed [71], transforms it using the activation function *g*, and returns the transformed output to any output connections.

The MLP classifier is an eager learning-based classification approach belonging to the discriminative category of classifiers. At its core is a feed-forward Neural Network with nodes arranged into three or more layers including an input layer containing all input nodes, an output layer containing all output nodes, and one or more hidden layers, which sit between the input and output layers. Within each layer every node is fully connected with all nodes in the neighbouring layers, but no connections are made between the nodes of a single layer or to nodes in non-neighbouring layers. All connections between nodes contain the product of output returned by the node feeding into the connection and a weight value, which is determined during the training of the network. Furthermore, aside from input nodes, which allow data input to pass directly through them, all other nodes in the MLP network sum and pass their input data through a non-linear activation function. For our research, we have chosen to assess the MLP architecture demonstrated in figure 6.5. This architecture, which reflects the one used in [32], contains three layers and uses the logistic sigmoid function (equation 6.4) as its activation function. Each node in the input layer corresponds with an individual feature from the feature space used to train the MLP, while each output node corresponds with a different subject from the training data. With this structure in place, the output for a particular subject node will be a value between 0 and 1, signifying the confidence for which the network has determined any given input features match those previously learned for the given subject.

$$g(x) = \frac{1}{1+ e^{-x}}$$

Equation 6.4: The logistic sigmoid function used as our neural network activation function.

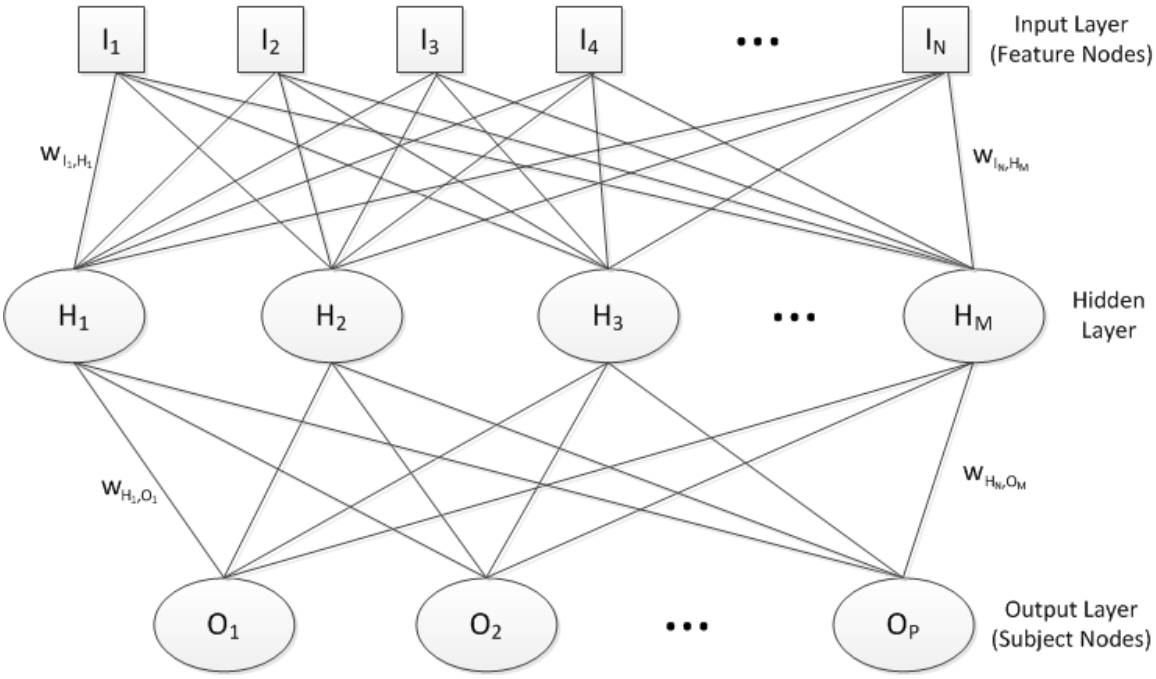

Figure 6.5: This diagram demonstrates our chosen 3 layer MLP architecture. The nodes in the input layer contain no activation function and pass any received data direct into the connection with the hidden layer, while the hidden and output nodes were configured to use a logistic sigmoid activation function. Each node is connected with every node in its immediate neighbours and all connections apply a weight to the data going into the following layer, as demonstrated on the edge connections.

The MLP architecture described to this point gives us the means to learn how to identify subjects based on their features, but before it can perform any classifications it must first undergo a training process. As previously mentioned, the data passed between connected nodes in the MLP is multiplied by weight values. These weight values can be adjusted so to minimize the error rate produced by the classifier for a set of training input-output pairs via a process known as back-propagation [72]. The back-propagation process is premised on the fact that the MLP network can be represented as a vector function $\boldsymbol{h_w(x)}$ parameterized by its weights ($\boldsymbol{w}$). So, if we had a pair of input features ($\boldsymbol{x}$) and an output

subject set (***y***) with a value of 1 corresponding to subject nodes that match the input and 0 for those that do not, then the vector of errors for our network's output nodes would be represented as $\boldsymbol{E} = \boldsymbol{y} - \boldsymbol{h_w}(\boldsymbol{x})$. Using this set of error values, it is possible to derive the network's node deltas, which, for any given connection from node *i* to node *j*, would represent the effect of a change in the input to node *j* on the output of node *i* [73]. At the output layer these node deltas can be calculated directly, while in the input and hidden layers they are computed by back-propagating the values of the deltas computed in the deeper neighbouring layer, as demonstrated in equation 6.5. These node deltas can also be used in conjunction with the computed output from the node feeding into the delta connection to derive the partial derivative (gradient) for each of the network's weights with respect to the error value (equation 6.6). Consequently, to minimize the error value we can use the gradient descent method, which follows the contour of the error surface in the direction of steepest descent [73].

$$\boldsymbol{E} = \boldsymbol{y} - \boldsymbol{h_w}(\boldsymbol{x})$$

$$in_i = \sum_j w_{ij} a_j + bias_i$$

$$\delta_i = \begin{cases} E_i \times g'(in_i) & , \ output\ nodes \\ g'(in_i) \times \sum_k \delta_k \times w_{ki} & , \ input\ /\ hidden\ nodes \end{cases}$$

Equation 6.5: These equations demonstrate the derivation of the node deltas. In this equation the term in_i represents the sum of the weighted outputs from node *i* to node *j*, which are to be passed back to the derivative of *i*'s activation function. The node deltas themselves for each node *i* are identified as δ_i, w_{ij} represents the weight between nodes *i* and *j*, and E_i the portion of the error produced by a given output node *i*.

$$\frac{\partial E}{\partial w_{ij}} = \delta_i \times a_j$$

Equation 6.6: This equation demonstrates the gradient of the error with respect to the weight between nodes *i* and *j*, at the point for which the output of node *j*'s activation function is a_j.

The use of gradient descent in the back-propagation process involves iteratively adjusting the network weights via the weight update function shown in equation 6.7. This process will continue for a predefined number of iterations or until the returned error falls below some maximum threshold, at which point the network error term can be assumed to have been brought to a local minimum. In our chosen weight update function, the speed at which the error is minimized is subject to two constants, namely the learning rate ($\in$) and momentum (α). The learning rate can be used to adjust the size of the step taken down the gradient for each weight update, while the momentum term prevents the weight updates from oscillating between opposing sides of a trough in the error contour by taking into account the trajectory of the previous iteration [73]. Additionally, the weight update function can be altered so that it minimizes the error across multiple training samples during a single update by batching (summing) the gradients produced by each individual training sample (equation 6.8). It is this batching-based weight update function that represents the final piece of the process needed to train our MLP classifier with our multiple GRF feature-subject pairs.

$$\Delta w_{ij}(n+1) = \epsilon(\delta_i \times a_j) + \alpha \Delta w_{ij}(n)$$

Equation 6.7: This equation demonstrates the standard weight update function, which is used to iteratively update the weights such that they minimize the network error term. In

this equation ϵ is a constant representing the back-propagation process's learning rate, while α is a constant accounting for momentum.

$$\Delta w_{ij}(n+1) = \epsilon \sum_p (\delta_{ip} \times a_{jp}) + \alpha \Delta w_{ij}(n)$$

Equation 6.8: This equation demonstrates the batched weight update function, a variant of the weight update function that allows the process to simultaneously reduce the error for all input-output pairs *p* provided at the time of training.

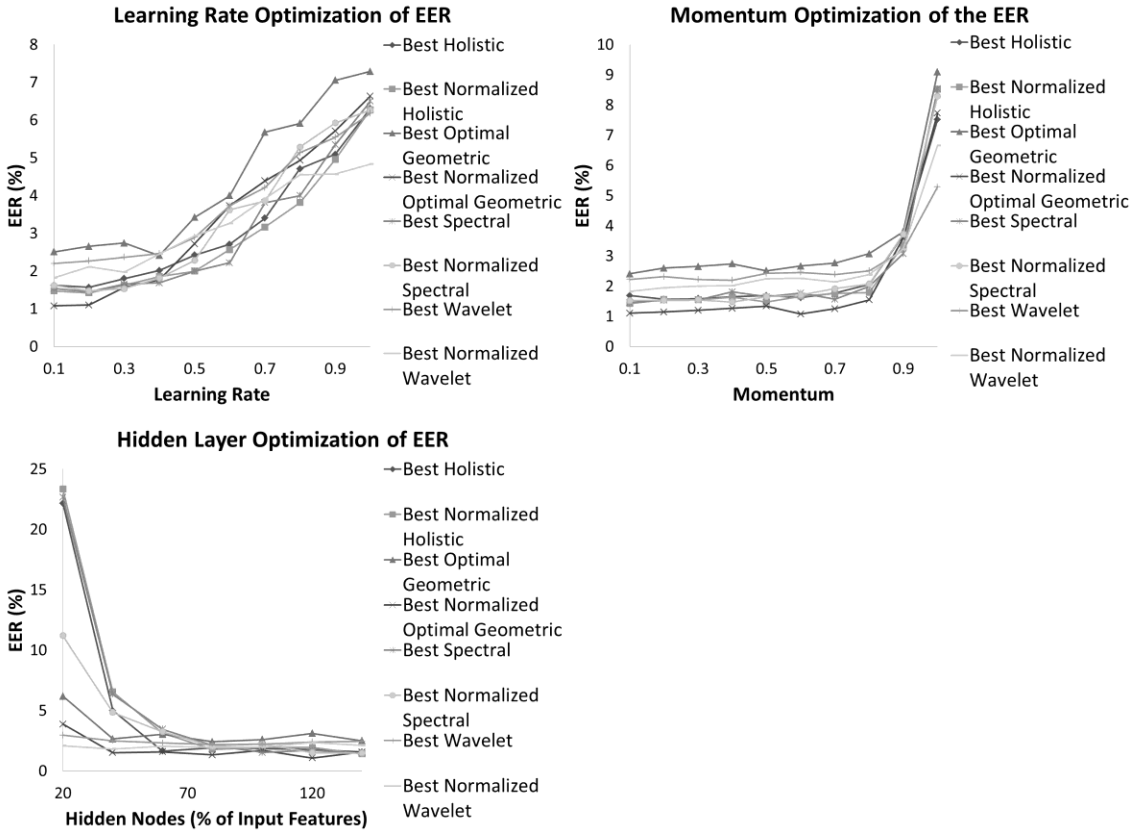

Figure 6.6: This figure compares our best cross-validated EER values achieved with the optimization parameters used to achieve them for all of our best performing preprocessing techniques. For any given parameter value, optimization was carried out by testing the given parameter value with every possible value for the other two parameters and returning the EER for the best combination.

Our implementation of the MLP was constructed using the Encog Machine Learning Framework for C# [74]. Following the typical convention for MLP weight initialization, we randomized our weights prior to performing back-propagation [72], making use of a seeded randomizer with a standard seeding to ensure consistency between training runs. We also configured our training process to terminate after either completing 100,000 iterations or when the network's computed error term fell below a value of 0.0000001. This left four configurable inputs to be accounted for: the input features, the learning rate, the momentum, and the number of hidden nodes. As discussed in the previous section, undesirable bias in input features can be mitigated by rescaling each feature space to a common scale. For the KNN classifier this was accomplished by rescaling the input features to fall within the range [0, 1]. In our MLP classifier we instead went with a rescaling range of [-1, 1], as our chosen activation function, the logistic sigmoid function, expects input to be distributed around the zero mark. The use of the sigmoid activation function in our network also meant the results produced at our output nodes came out as values between 0 and 1, representing the network's confidence as to whether or not any given input features matched the subject corresponding to any given output node. Thus, unlike in the KNN classifier, we were able to use the output directly in our EER calculations, rather than first needing to find posterior probabilities. Having established a means to calculate our EER values, the three remaining configurable inputs were optimized via exhaustive search, whereby we tested the system with every combination of 10 different learning rate and momentum terms (the 10 evenly spaced points from 0.1 to 1.0) and 7 different hidden field sizes (the 7 evenly spaced nearest integers from 20% to 140% of input feature space size). In total, this meant testing 700 different

combinations of parameters for each of our best performing preprocessors. The best achieved GRF recognition results for each tested configuration of learning rate, momentum, and hidden nodes are demonstrated in figure 6.6.

Looking back at the optimization results in figure 6.6, we found our best GRF recognition performance was achieved with MLP learning rate terms less than 0.5, momentum terms less than 0.8, and with the hidden nodes numbering over 60% the size of input feature nodes. The best of these results, broken down by feature extractor, are demonstrated in table 6.2. In contrast with the findings of [32], in which the MLP classifier led to a 9.4% improvement in GRF geometric feature space recognition over the KNN classifier, we noticed a substantial decline in our recognition performance with the application of the MLP classifier to our optimal geometric feature spaces. Conversely, the performance increase we achieved with the application of the MLP classifier to our spectral features was in agreement with the findings of [32]. The results, as a whole, showed a clear increase in GRF recognition performance when the MLP classifier was applied to feature spaces derived via unsupervised PCA, and a clear decrease in performance when it was applied to feature spaces derived using supervised learning approaches. A likely explanation for these discrepancies points to the nature of the feature extraction techniques derivation. Each feature space was optimized to some degree using the KNN classifier, leading to an inherent performance bias toward the KNN algorithm used for feature space optimization. This bias would have been far greater in feature spaces derived via supervised learning, thus outweighing any potential performance increases that would otherwise be achieved using the MLP classifier. Nevertheless, our MLP

classifier overcame this inherent bias in the unsupervised feature spaces, which would seem to suggest that the MLP would lead to better recognition performance than the KNN classifier if it were to be used in place of the KNN classifier for feature space optimization.

Optimal MLP Classifier Results

Pre-processor	Learning Rate	Momentum	Hidden Nodes	Threshold	Cross Validated EER (%)	EER Improvement (%)
Best Optimal Geometric	0.4	0.1	29	0.05	2.41111	-80.8
Best Normalized Optimal Geometric	0.1	0.6	66	0.1656	1.07777	-506.2
Best Holistic	0.2	0.2	21	0.1125	1.56666	38.6
Best Normalized Holistic	0.2	0.1	19	0.3	1.42222	30.4
Best Spectral	0.2	0.1	19	0.0906	1.45555	28
Best Normalized Spectral	0.2	0.4	24	0.1062	1.47777	19.8
Best Wavelet	0.1	0.4	64	0.2968	2.2	-70.6
Best Normalized Wavelet	0.1	0.1	39	0.2	1.82222	-65.6

Table 6.2: This table demonstrates the best performance achieved by the MLP classifier for each preprocessing technique. The threshold shown is the threshold at which the EER value was calculated (a value between 0 and 1 derived from the raw MLP output) and the EER improvement represents improvement in recognition performance achieved by the optimal MLP variant over the results calculated in the previous two chapters.

6.3 Support Vector Machine

In the previous section we set out to demonstrate the claim that a MLP classifier could be used to derive class-separating boundaries not otherwise obtainable using a KNN classifier. The results obtained after applying the MLP classifier to our GRF data appeared to support this claim. However, research into MLPs has shown that they are prone to producing boundaries so tightly aligned with the specific characteristics of their training samples' class distribution that they may be unable to produce the strong generalization of boundaries needed to account for variability in the unseen test samples [75, 76]. One classification model that, in literature, has often been found to produce better class boundary generalization is the Support Vector Machine (SVM) [77]. The SVM-based classification technique was the second most widely used classification technique in our previously examined GRF-recognition studies [7, 32, 3, 4], and in each of these studies it produced the best recognition performance when compared with all other studied classifiers. This discriminative classifier is an eager learner, in that it performs classification optimizations as part of a distinctive training phase; however, it could also be considered to have properties of an instance-based learning algorithm, in that it retains a subset of samples from the original training dataset to assist with the definition of boundaries during its usage. At the core of the SVM-based classification technique are one or more binary linear classifiers (SVMs), which, when grouped together, can be used to solve multiclass problems [78]. Each individual SVM is derived via solving for the maximum-margin hyperplane separating a provided two-class training dataset.

To understand the SVM maximum-margin problem consider the linear separator dividing the samples from two different classes in the two dimensional feature space shown in figure 6.7. Although this linear separator could have been drawn in numerous locations and still have accomplished a separation of classes, the chosen location, which represents the maximum distance between the separator and nearest samples of either class, was selected because research has shown it often provides strong generalization boundaries [79]. Consequently, it follows that by solving for the line that forms this maximum-margin separator we would have a high likelihood of acquiring a set of boundaries that separate the samples of our different GRF subjects better than was possible with the classification methods of the previous two sections. To solve for the maximum margin separator we first need to frame the problem such that our linear separator is defined with respect to our training samples, which we will refer to as the feature vectors $\boldsymbol{x_i}$ for any sample *i*, and with respect to the labels of the two classes that we are attempting to divide, which we will identify as +1 for one class and -1 for the other ($y_i \in \{+1, -1\}$). Looking again at figure 6.7 we can see that we have bounded the linear separator ($\boldsymbol{w}^T\boldsymbol{x} + b = 0$) between the two parallel lines representing the margin boundaries ($\boldsymbol{w}^T\boldsymbol{x} + b = 1$ and $\boldsymbol{w}^T\boldsymbol{x} + b = -1$) that run through the points sitting nearest to the separator for both classes and satisfy the constraints in equation 6.9. At this point we know the values for $\boldsymbol{x}$ as well as the values of $\boldsymbol{w}^T\boldsymbol{x} + b$ for the vector points that sit on the margin boundaries (referred to as the support vectors). If we were to calculate the distance from the separator to any point on either of the parallel margin boundaries we would find that the margin separating the two classes can be maximized by minimizing the Euclidean norm of our normal vector ($\|\boldsymbol{w}\|$), as demonstrated in equation 6.10. This

minimization problem, bounded by the constraints defined in equation 6.9, is typically referred to as the primal optimization problem for SVM.

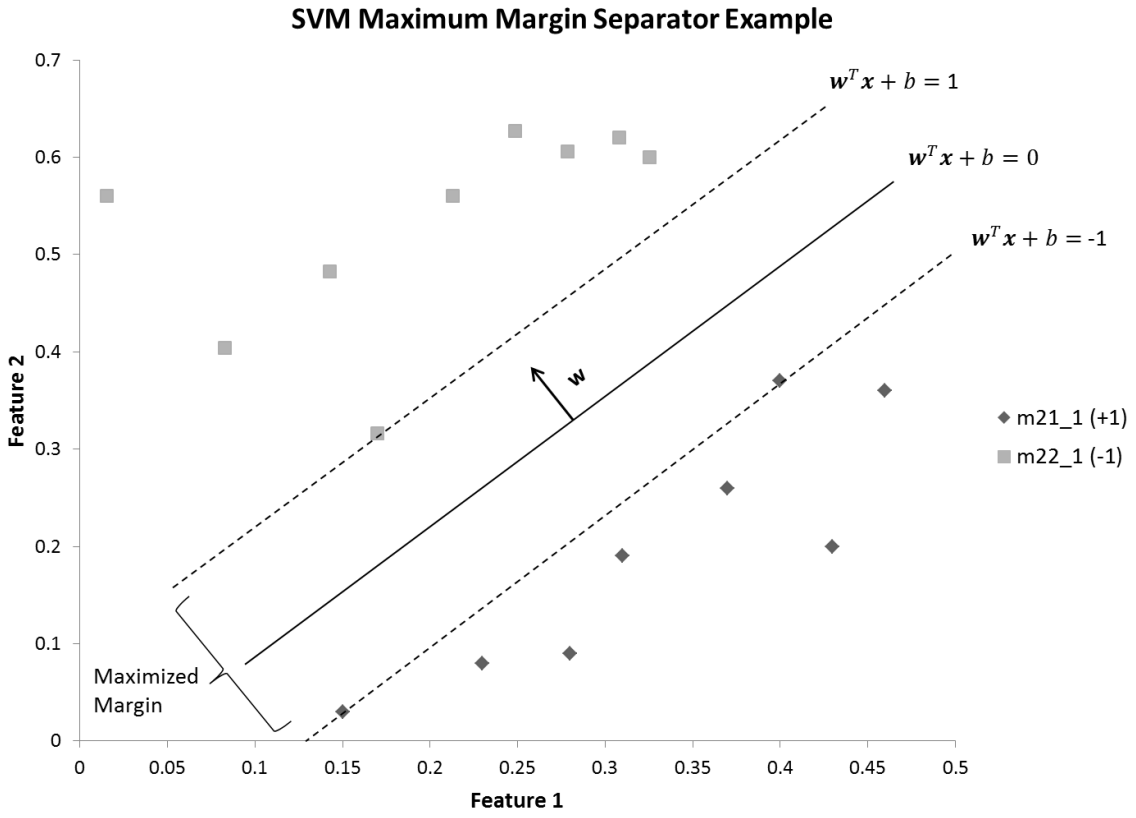

Figure 6.7: This figure demonstrates the separation of the samples for two different GRF subjects using the maximum margin hyperplane $\boldsymbol{w^T x + b = 0}$. In this example, the samples are represented in a 2-dimensional feature space, yet the separator is defined such that it would still be applicable for a space containing any number of dimensions.

$$\boldsymbol{w^T x_i + b \geq 1} \quad for \quad y_i = +1$$

$$\boldsymbol{w^T x_i + b \leq -1} \quad for \quad y_i = -1$$

$$=> \quad y_i(\boldsymbol{w^T x_i} + b) \geq 1 \quad or \quad -y_i(\boldsymbol{w^T x_i} + b) + 1 \leq 0 \quad \forall_i$$

Equation 6.9: This equation describes the constraints bounding the linear definition of the SVM margin in figure 6.7, where x_i refers to the sample value vectors and y_i refers to the

sample class labels for any training sample *i*. Defining the class labels as 1 and -1 allows for the simplification of the two constraints into a single constraint.

h_1: $\boldsymbol{w}^T\boldsymbol{x_i} + b = 1$, h_2: $\boldsymbol{w}^T\boldsymbol{x_i} + b = -1$

$dist(\boldsymbol{w}^T\boldsymbol{x_i} + b = 0, \boldsymbol{x_0}) = \frac{|\boldsymbol{w}^T\boldsymbol{x_0}+b|}{\|\boldsymbol{w}\|}$, $\boldsymbol{x_0} \in h_1 \vee \boldsymbol{x_0} \in h_2$

$=> \quad dist(\boldsymbol{w}^T\boldsymbol{x_i} + b = 0, \boldsymbol{x_0}) = \frac{|\boldsymbol{1}|}{\|\boldsymbol{w}\|}$

$=> \quad dist(\boldsymbol{w}^T\boldsymbol{x_i} + b = 1, \boldsymbol{w}^T\boldsymbol{x_i} + b = -1) = \frac{2}{\|w\|}$ *(SVM margin)*

Primal Optimization Problem (margin maximization):

$\min_{w,b} \frac{1}{2}\|w\|^2 \quad subject\ to \quad -y_i(\boldsymbol{w}^T\boldsymbol{x_i} + b) + 1 \leq 0 \quad \forall_i$

Equation 6.10: This equation demonstrates the derivation of the SVM primal optimization problem. By calculating the distance between our two margin lines h_1 and h_2 under the constraints of equation 6.9, it can be seen that when the Euclidean norm of the normal vector is minimized the margin will be maximized (Note: minimizing half of its square can improve computational efficiency without changing the result).

The equations above demonstrated how the SVM maximum-margin problem can be presented as the optimization of a function bounded by constraints; alternatively, the problem can be redefined using Lagrangian multipliers ($\boldsymbol{\alpha}$) for the optimization of a single unbounded auxiliary function. Consider the Lagrangian auxiliary function defined in equation 6.11. If this function were to be maximized with respect to its Lagrangian multipliers then the resulting optimization would be found to produce an infinitely large value when the original constraints are not satisfied, but would produce a value equivalent to objective function ($\frac{1}{2}\|w\|^2$) when the original constraints are satisfied (demonstrated in equation 6.12). Knowing this, it can be shown that by minimizing the

function in equation 6.12 with respect to $\boldsymbol{w}$ and b (finding $\min_{\boldsymbol{w},b} \mathcal{L}_P(\boldsymbol{w}, b)$) we end up with a problem with an equivalent solution to that of the primal optimization problem. This form of the optimization problem is often referred to as the Lagrangian primal problem, and, with a bit of extra derivation it can also be shown that there is a dual form representation of this Lagrangian problem which further simplifies the computation required to find the solution to the primal optimization problem [80].

$$\mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{l} \alpha_i y_i (\boldsymbol{w}^T \boldsymbol{x_i} + b) + \sum_{i=1}^{l} \alpha_i$$

Equation 6.11: This equation demonstrates how Lagrangian multipliers ($\boldsymbol{\alpha}$) can be combined with the function being minimized and the left side of the constraint inequality from the optimization problem defined in equation 6.10. In this derivation *l* represents the number of different samples being optimized.

$$\mathcal{L}_P(\boldsymbol{w}, b) =$$

$$\max_{\boldsymbol{\alpha}:\, \alpha_i \geq 0} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \begin{cases} \frac{1}{2}\|\boldsymbol{w}\|^2 & if\ \boldsymbol{w}\ satisfies\ the\ primal\ constraints \\ \infty & otherwise \end{cases}$$

Equation 6.12: By maximizing the Lagrangian multipliers in equation 6.11 it can be shown that the part of the auxiliary function derived from the Primal Optimization Problem constraints may become infinitely large when the constraints are not satisfied (i.e. $-\boldsymbol{y_i}(\boldsymbol{w^T x_i} + \boldsymbol{b}) + \mathbf{1} > \mathbf{0}$); however, when they are satisfied the maximum value for the constraint part of the auxiliary function will be 0, thus giving $\boldsymbol{\mathcal{L}_P}(\boldsymbol{w}, \boldsymbol{b})$ the value of the function being optimized in equation 6.10.

The dual form of the Lagrangian problem (or Lagrangian dual problem) derives from the concept of duality, whereby there exist two related optimization problems, a primal and a dual problem, with the solution to the dual problem forming the lower bound to the

solution of the primal problem. While the solution to the dual problem always forms the lower bound to the primal problem, in certain circumstances they may share a common optimal solution. In the SVM case, when the relationship between the primal (p^*) and dual (d^*) problems in equation 6.13 is considered, it can be shown that these problems satisfy the necessary conditions for equality in their solution [80]. Thus solving the dual problem will also reveal the solution to the maximum margin separator of the primal problem. Moreover, because the auxiliary function that is being optimized (equation 6.11) is convex with respect to $\boldsymbol{w}$ and b for any fixed value of $\boldsymbol{\alpha}$, by taking the partial derivatives of $\boldsymbol{w}$ and b at the optimal value it is possible to derive the properties shown in equation 6.14. Using the discovered properties, the SVM Lagrangian dual problem's auxiliary function can be simplified such that it need only be optimized for the single variable $\boldsymbol{\alpha}$, as shown in equation 6.15. This reveals a form of the maximum margin problem that can easily be solved via quadratic optimization [81]. From this point it is then possible to derive the value of b for the maximum margin separator as the midway point between the values it takes on the two parallel margins, as shown in equation 6.16. Finally, the derived parameters can be substituted into equation 6.17 to form the desired SVM binary classifier (the foundation of the multi-class SVM classification technique), and, because it is also possible to show that the SVM dual form problem satisfies the Karush-Kuhn-Tucker complementary condition [77], it can also be shown that $\alpha_i = 0$ for any training vector point $\boldsymbol{x_i}$ that does not fall on the margin; thus, when performing classification via equation 6.17, classifier memory usage can often be optimized, as only support vectors (typically a small subset of training samples) are needed to derive the classification boundary and all other values are ignored.

$$d^* = \max_{\boldsymbol{\alpha}:\, \alpha_i \geq 0} \min_{\boldsymbol{w},b} \mathcal{L}(\boldsymbol{w}, b, \alpha) \quad \leq \quad \min_{\boldsymbol{w},b} \max_{\boldsymbol{\alpha}:\, \alpha_i \geq 0} \mathcal{L}(\boldsymbol{w}, b, \alpha) = p^*$$

Equation 6.13: This equation demonstrates the intuitive relationship between the SVM primal and dual problems. It has been shown that, given the functions being solved in this case, the primal problem is in fact equal to the dual problem [80] so $\boldsymbol{d^* = p^*}$.

$$\frac{\partial}{\partial \boldsymbol{w}} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \boldsymbol{w} - \sum_{i=0}^{l} \alpha_i y_i \boldsymbol{x_i} = 0 \implies \boldsymbol{w} = \sum_{i=0}^{l} \alpha_i y_i \boldsymbol{x_i}$$

$$\frac{\partial}{\partial b} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i y_i = 0$$

Equation 6.14: The relationships shown in this equation are derived by minimizing $\boldsymbol{\mathcal{L}(w, b, \alpha)}$ with respect to *w* and *b*. The optimal minimum value occurs when the partial derivatives are equal to zero since the function $\boldsymbol{\mathcal{L}(w, b, \alpha)}$ is convex with respect to both parameters for any value of $\boldsymbol{\alpha}$.

$$\mathcal{L}_D(\boldsymbol{\alpha}) = \min_{\boldsymbol{w},b} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha})$$

$$= \frac{1}{2}\left(\sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x_i}\right)^T \left(\sum_{j=1}^{l} \alpha_j y_j \boldsymbol{x_j}\right) - \sum_{i=1}^{l} \alpha_i y_i \left(\sum_{j=1}^{l} \alpha_j y_j \boldsymbol{x_j}\right)^T \boldsymbol{x_i} - b \sum_{i=1}^{l} \alpha_i y_i +$$

$$\sum_{i=1}^{l} \alpha_i$$

$$= \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \boldsymbol{x_i}^T \boldsymbol{x_j}$$

$$d^* = \max_{\boldsymbol{\alpha}} \mathcal{L}_D(\boldsymbol{\alpha}) \qquad subject\ to \quad \alpha_i \geq 0,\ i = 1, \ldots, l \quad and \quad \sum_{i=1}^{l} \alpha_i y_i = 0$$

Equation 6.15: This equation demonstrates the derivation of the simplified SVM Lagrangian dual problem obtained by substituting the findings from equation 6.14 into the auxiliary function of equation 6.11. Note that the function now requires only a single parameter be optimized.

$$b = -\frac{\max_{i:y_i=-1} \boldsymbol{w^T x_i} + \min_{i:y_i=1} \boldsymbol{w^T x_i}}{2}$$

Equation 6.16: This equation demonstrates the derivation of *b* using the discovered values of *w*. In this equation the maximum value for points identified by $\boldsymbol{y_i = -1}$ will sit on one margin while the minimum value for those identified by $\boldsymbol{y_i = 1}$ will sit on the other, so by

taking one half of the combined *b* component for each margin it is possible to obtain its value at the separator.

$$\boldsymbol{w^T x} + b = \left(\sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x_i}\right)^T \boldsymbol{x} + b = \sum_{i=1}^{l} \alpha_i y_i \boldsymbol{x_i}^T \boldsymbol{x} + b$$

Equation 6.17: This equation demonstrates how the derived maximum margin separator can be used to perform classification for a given vector point *x* and training points 1 … *l*. If the resulting value is greater than zero then the vector *x* is classified as belonging to the class which was given the label $\boldsymbol{y_i = 1}$, otherwise it is classified as belonging to the class given the label $\boldsymbol{y_i = -1}$.

The SVM classifier as it has been defined to this point has a couple of weaknesses that make it poorly suited to more complicated classification tasks, namely it depends upon its training data being linearly separable and it can be susceptible to undesirable influence from outlier points. To address these problems a slightly different variant of SVM known as soft margin SVM can be used. In soft margin SVM training samples are allowed to sit within the margin or even on the wrong side of the separator. This is accomplished by redefining the margin constraint as $1 - \xi_i$, where the non-negative slack variables ξ_i represent the degree to which a given vector point $\boldsymbol{x_i}$ is sitting on the wrong side of its class's margin boundary. In this case the primal optimization problem becomes the simultaneous maximization of the margin separating the two classes and minimization of the degree to which training points are situated on the incorrect side of the margin to achieve the respective maximum margin. In the formal definition of the soft margin SVM's primal optimization problem (equation 6.18) a new regularization parameter (*C*) is introduced, which allows for adjustments to the relative weighting given to the two problem components. From here, using an additional Lagrangian multiplier ($\beta \geq 0$), the

Lagrangian function of equation 6.11 can be redefined to reflect the inclusion of the new slack variables as shown in equation 6.19. To get the dual form of this new problem the partial derivative of the new Lagrangian function can be taken with respect to ξ, in addition to the partial derivatives of $\boldsymbol{w}$ and b shown earlier in equation 6.14; this in turn results in a new dual problem (equation 6.20) that only differs from equation 6.15 in the definition of the constraint placed on $\boldsymbol{\alpha}$. Under this new problem definition the support vectors include not only the trial sample points that sit on the margins, but also trial samples that sit on the wrong side of the margins. Additionally, when using the soft margin form of SVM, the new technique shown in equation 6.21 must be used to solve for b, as opposed to the technique previously demonstrated in equation 6.16. Yet, having solved for its parameters, SVM classification can once again be accomplished using equation 6.17 via substituting in the new soft-margin solutions for $\boldsymbol{\alpha}$ and b.

$$\min_{\boldsymbol{w},\xi,b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i \qquad subject\ to \qquad -y_i(\boldsymbol{w}\boldsymbol{x_i} + b) + (1 - \xi_i) \leq 0 \qquad \forall_i$$

Equation 6.18: This equation demonstrates the soft margin SVM primal optimization problem. In this equation samples are allowed to sit on the wrong side of their margin boundaries; however, in doing so the degree to which each is misclassified ($\boldsymbol{\xi_i \geq 0}$) incurs a penalty cost in the minimization of the objective function. The regularization parameter *C* is used to adjust the weight of which the penalty is applied.

$$\mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}\xi_i - \sum_{i=1}^{l}\alpha_i(y_i(\boldsymbol{w^T x_i} + b) - (1 - \xi_i)) - \sum_{i=1}^{l}\beta_i\xi_i$$

Equation 6.19: This equation demonstrates the Lagrangian representation of the soft margin SVM objective function. It is similar to equation 6.11; however, it includes the addition of slack variables ($\boldsymbol{\xi}$) and their respective Lagrangian multipliers ($\boldsymbol{\beta}$).

$$\frac{\partial}{\partial \boldsymbol{\xi}} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = -\sum_{i=1}^{l} \alpha_i + C - \sum_{i=1}^{l} \beta_i \Rightarrow \alpha_i = C - \beta_i \qquad \forall_i$$

$$\alpha_i \geq 0,\ \beta_i \geq 0 \Rightarrow C - \alpha_i \geq 0 \text{ and } \alpha_i \geq 0 \Rightarrow \alpha_i \leq C \text{ and } \alpha_i \geq 0$$

$$\mathcal{L}_D(\boldsymbol{\alpha}) = \min_{\boldsymbol{w},\boldsymbol{\xi},b} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j \boldsymbol{x_i}^T \boldsymbol{x_j}$$

$$d^* = \max_{\boldsymbol{\alpha}} \mathcal{L}_D(\boldsymbol{\alpha}) \qquad subject\ to \quad 0 \leq \alpha_i \leq C,\ i = 1, \ldots, l \quad and \quad \sum_{i=1}^{l} \alpha_i y_i = 0$$

Equation 6.20: In this equation it is shown that the addition of the slack variable to the soft margin dual problem only has the effect of altering the constraint on $\boldsymbol{\alpha}$ from Lagrangian dual problem previously defined in equation 6.15.

$$b = \frac{1}{|\boldsymbol{S}|} \sum_{i \in \boldsymbol{S}} \left(y_i - \sum_{j \in \boldsymbol{S}} \alpha_i y_i \boldsymbol{x_i}^T \boldsymbol{x_j} \right)$$

Equation 6.21: This equation demonstrates the solution to *b* for soft margin SVM, with *S* being the subset of training points that form the support vectors (i.e. the points sitting on the margin boundaries together with those sitting on the wrong side of the margin boundaries).

The application of soft margin SVM can be useful when solving problems that are nearly, but not quite, linearly separable, yet it will fail when applied to problems that are clearly not linearly separable, such as the one on the left side of figure 6.8. To get around this limitation SVMs use a technique known as the kernel trick, which is premised on transforming a non-linearly separable feature space into a higher dimensional feature space that can more easily be separated linearly (for instance the feature space on the right side of figure 6.8). Under the kernel trick the function $\emptyset(\boldsymbol{x})$ is defined as a function that transforms a sample $\boldsymbol{x}$ into a higher dimensional feature space. If this function were applied to the two sample points in the Lagrangian dual problem of equation 6.20 then, rather than searching for a linear separator in our original feature space, we would instead be searching for a linear separator in the high dimensional space defined by $\emptyset$. In doing

so the application of the dimension expansion function would result in the dot product of the original dual problem equation $\boldsymbol{x_i}^T \boldsymbol{x_j}$ becoming $\emptyset(\boldsymbol{x_i})^T \emptyset(\boldsymbol{x_j})$, which in many cases can be efficiently computed without ever having to compute $\emptyset$ for $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ individually [77]. The function that accomplishes this simplified computation of the dot product over a higher dimensional space is known as the kernel function and is typically denoted as $\kappa(\boldsymbol{x_i}, \boldsymbol{x_j})$. To use the kernel trick with soft margin SVM, the chosen kernel function would be substituted to replace of any dot products of the input samples $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ within the Lagrangian dual problem (equation 6.20), the calculation of *b* (equation 6.21), and the binary SVM classifier (equation 6.17). As for the definition of the kernel function itself, there are many valid forms its equation could take but research has shown that one kernel in particular, the Gaussian Radial Basis Function (RBF) kernel (equation 6.22), outperforms most others when applied to the GRF recognition problem [4]

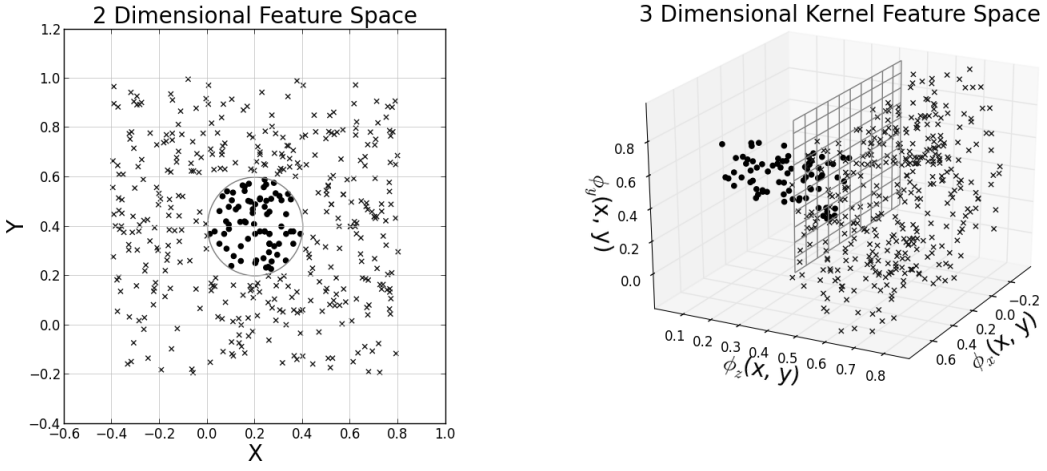

Figure 6.8: The two figures above demonstrate how a dataset that cannot be linearly separated in 2-dimensions (the diagram on the left) can be transformed into a higher dimensional kernel space that can be separated linearly (the diagram on the right).

$$\kappa(\boldsymbol{x_i},\ \boldsymbol{x_j}) = \ \emptyset(\boldsymbol{x_i})^T \emptyset(\boldsymbol{x_j}) = \exp\left(-\gamma \left\| \boldsymbol{x_i} - \ \boldsymbol{x_j} \right\|^2\right)$$

Equation 6.22: This equation demonstrates the Gaussian Radial Basis Function for kernel SVM. This equation takes two point vectors as input and produces output equivalent to that which would be produced if the dot product was taken after both points were transformed into a higher dimensional space. The shape of the transform can be tuned to optimize the strength of the SVM separator by adjusting the parameter $\boldsymbol{\gamma}$.

Using a single soft-margin kernel-SVM gives us a powerful tool for classifying points in two-class problems; however, to perform SVM classification on any multi-class problem, including our GRF subject-recognition problem, we require multiple SVMs and a strategy to train them and evaluate their results. Two strategies are commonly used to accomplish multi-class classification with SVM: one-against-one and one-against-all. In the typical one-against-one strategy individual SVMs are first trained for each of the *C(C-1)/2* training class pairs taken from the *C* training classes, then, during classification the class that each SVM selects is given single vote and the sample being classified is classified as belonging to the class that gains the most votes; this strategy was used for GRF-recognition in [7]. A slightly different approach is taken in the typical one-against-all strategy; rather than deriving SVMs for every class pair, in one-against-all a SVM is first trained for each class against a grouping of all the samples in every other training class, then during classification the class with the highest raw SVM output value is assigned to the sample being classified. In their common form neither of the two multi-class SVM strategies output posterior probabilities, however, both can be manipulated to give such output [78]. A common approach, implemented in the popular LIBSVM tool [82], uses a modified version of the one-against-one strategy to calculate posterior probabilities. In this case, rather than treating the SVM's as binary classifiers with votes,

the unscaled raw output of each individual SVM output is passed into a sigmoid function (equation 6.23) to produce an estimate of the pairwise probabilities for each of the *C(C-1)/2* training class pairs. To estimate the values for *A* and *B* in equation 6.23 this implementation minimizes the negative log likelihood of the training data [83]. And, having found the pairwise probabilities, the probability (p_i) that a given tested sample $\boldsymbol{x}$ belongs to a given training class *i* can be estimated by solving for the optimization problem in equation 6.24, as discussed by Wu et al. in [84].

$$r_{ij} \approx P(y = i | y = i \; or \; j, \boldsymbol{x}) \approx \frac{1}{1+\exp(Af_{ij}(\boldsymbol{x})+B)}$$

Equation 6.23: This equation demonstrates the sigmoid function that can be used to estimate the probability that a given sample *x* belongs to class *i* for the output of an SVM trained across classes *i* and *j*. The values for *A* and *B* are calculated separately by minimizing the log likelihood function [83].

$$\min_{\boldsymbol{p}} \frac{1}{2}\sum_{i=1}^{k}\sum_{j:j=1,j\neq i}^{k}\left(r_{ji}p_i - r_{ij}p_j\right)^2 \quad subject\ to \quad p_i \geq 0, \forall_i, \quad \sum_{i=1}^{k} p_i = 1$$

Equation 6.24: This equation demonstrates how the pairwise probabilities (*r*) calculated in equation 6.23 can be incorporated into an optimization problem to solve for the probability (*p*) that the sample used to generate the pairwise probabilities belongs to each of the training data classes.

To this point we have demonstrated the probabilistic multi-class soft-margin variant of kernel SVM previously used in [32]. This is the configuration we decided upon using for our own implementation of SVM for GRF recognition, which we developed using the Encog [74] wrapper of the popular LIBSVM tool [82]. For our kernel function we opted for the Gaussian RBF function as this was selected as the kernel function in all of the

previously SVM-based GRF recognition studies. To effectively generate EER values using this technique we created an extension of the standard Encog SVM class, constructing it to initialize with the LIBSVM *probability* parameter flag set to 1. Furthermore, we altered the default LIBSVM probability generation behaviour, which, due to the pseudo-random cross validation used in its probability generation algorithm, effectively returned non-deterministic probabilities. In our implementation we had the LIBSVM code use deterministic probabilities by setting Encog's *SupportClass.Random* value to use a seeded random number rather than the default pseudo-random number. Finally, we optimized our SVM classifier with respect to its input parameters: the features passed in for training and testing, the value of the regularization parameter (*C*), and the value of the kernel parameter (γ). First, to reduce the undesirable bias from any particular input feature we rescaled all our training and testing input features to fall within the range of [0, 1], as defined by the respective minimum and maximum values for each feature within the training dataset. Having rescaled our input features, the values of our two additional SVM parameters were then optimized via an exhaustive search of various parameter combinations similar to the approach used in our MLP optimization. By picking a reasonable distribution of arbitrary parameter values we were able to identify the range of values for each parameter that optimized the GRF recognition EER results when used with each of our preprocessor configurations (shown in figure 6.9).

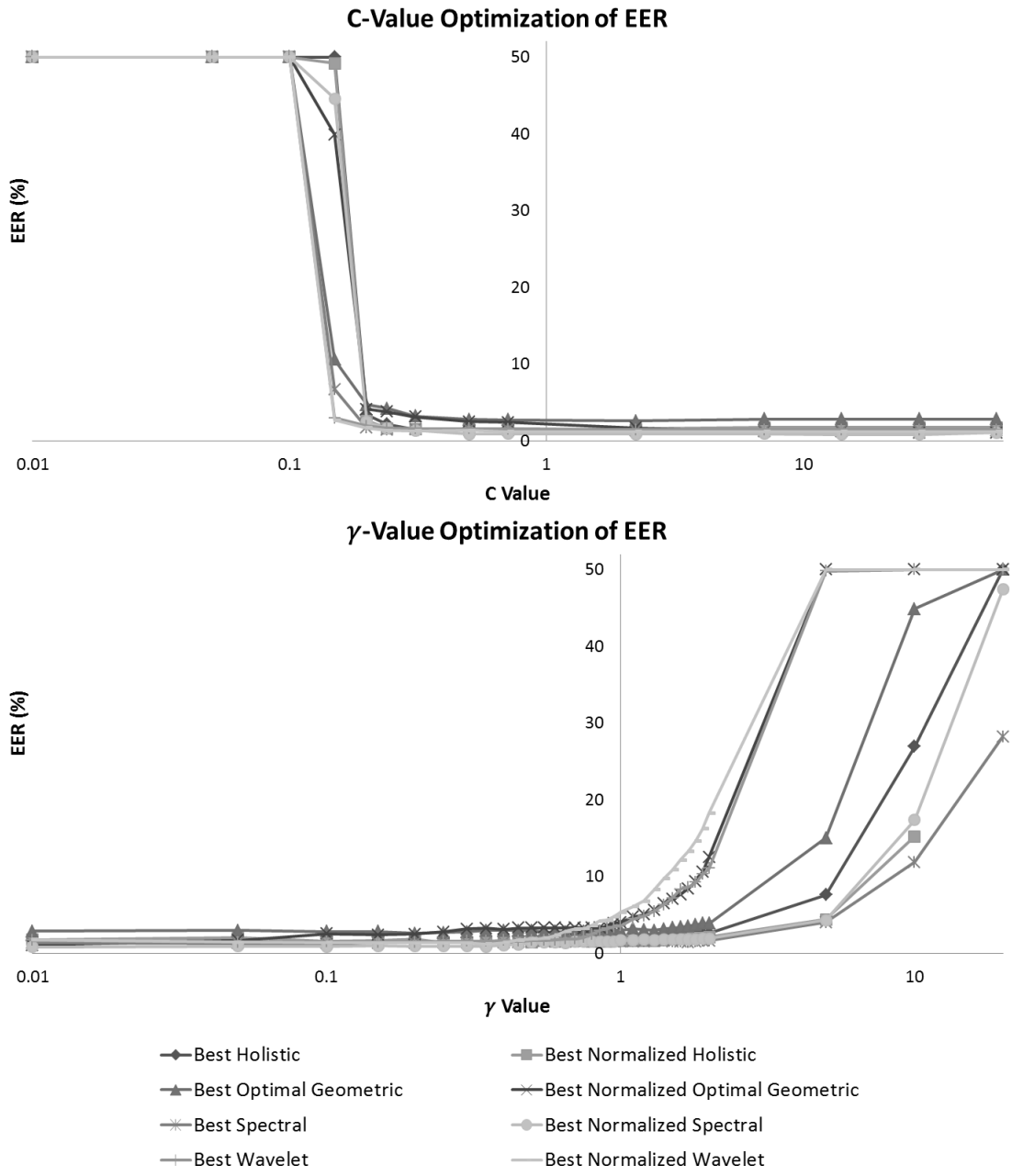

Figure 6.9: This figure compares our best cross-validated EER values achieved with the SVM classifier to the optimization parameters used to achieve them for all of our best performing preprocessing techniques. For either given parameter value, optimization was carried out by testing the given parameter value with every possible value for the other parameter and returning the EER for the best combination.

In figure 6.9 we can clearly see that the regularization parameter is ineffective for GRF recognition when its value is below about 0.25, thereafter having little influence on performance when given larger values. Conversely, the optimization of the kernel parameter achieved its best performance when given a value below 1 and quickly became ineffective when given larger values. The combination of parameters that produced the best GRF recognition results across our best feature extraction configurations are shown in table 6.3. As was the case with our MLP classifier, the use of a SVM classifier led to a substantial decrease in GRF recognition performance when used in combination with our optimal geometric and wavelet feature spaces, likely owing to their relatively strong inherent performance bias toward KNN (a result of the use of KNN in our feature extractor optimization). However, as was also the case with our MLP classifier, the GRF recognition performance increased for our two feature spaces that were less subject to KNN bias (our holistic and spectral feature spaces). In fact the use of the SVM classifier together with our holistic and spectral feature spaces actually led to a significant increase in GRF recognition performance when compared with the MLP classification results, most notably in the non-normalized holistic and normalized spectral feature spaces. These findings, when accounting for the KNN bias, support the findings of previous GRF recognition studies [7, 32, 3, 4], which found SVM improved recognition performance over the KNN classifier. Furthermore, our findings on the comparison of SVM with MLP reflected those of [32], with SVM generally achieving comparable or better GRF recognition performance than MLP across a variety of different feature spaces.

Optimal SVM Classifier Results

			Cross Validated		
Pre-processor	**C**	**γ**	**Threshold**	**EER (%)**	**EER Improvement (%)**
Best Optimal Geometric	2.24	0.2	0.1531	2.61111	-95.8
Best Normalized Optimal Geometric	56.56	0.01	0.2109	1.04444	-487.5
Best Holistic	2.24	0.1	0.1656	0.96666	62.1
Best Normalized Holistic	0.71	0.25	0.1531	1.38888	32
Best Spectral	7.07	0.05	0.1859	1.31111	35.1
Best Normalized Spectral	14.14	0.01	0.1984	0.83333	54.8
Best Wavelet	2.24	0.15	0.1593	1.43333	-11.2
Best Normalized Wavelet	0.71	0.35	0.1406	1.2	-9

Table 6.3: This table demonstrates the best performance achieved by the SVM classifier for each preprocessing technique. The threshold shown is the threshold at which the EER value was calculated (a value between 0 and 1 derived from the raw posterior probability output) and the EER improvement represents the improvement in recognition performance achieved by the optimal SVM variant over the results calculated in the previous two chapters.

6.4 Linear Discriminant Analysis

In section 2.2.3 of Chapter 2 we discussed the categorization of classifiers as following either a generative or discriminative model for establishing whether a given sample belongs to a specific class. The classifiers we have examined so far were all categorized as being discriminative classifiers because their posterior class probabilities were derived directly from their optimized outputs, without regard for the underlying class conditional densities. In this section we explore the use of the eager learning classifier known as Linear Discriminant Analysis (LDA), which, despite its name, takes a generative approach to modeling posterior class probabilities. Traditionally, in its basic form, this classifier assumes each class has a Gaussian distribution of members with a common degree of variance across all classes [85]. Under such assumptions posterior probabilities are derived by first finding the position of a tested sample with respect to each class's conditional multivariate Gaussian probability density function (equation 6.25), then using Bayes rule (equation 6.26) to estimate the posterior class probabilities. This technique also comes in a reduced form which involves first using Fisher's dimensionality reduction technique to project the training and testing samples down to some smaller set of dimensions maximizing the statistical separation of classes, and then using this more discriminative reduced feature space to perform the traditional LDA classification [85]. Hastie et al. have suggested that the reduced form of LDA can produce better classification performance than the non-reduced form [85]; however, it is also susceptible to a deficiency known as the small sample size (SSS) problem [86], which occurs when the number of dimensions in the feature set being classified is greater than the number of data samples used for training. With regards to previous GRF recognition

studies, only [7] previously used LDA for GRF recognition and no details were given on the actual variant of LDA used. For the purpose of our research we have elected to study the GRF recognition performance of a non-SSS susceptible variant of reduced LDA known as Uncorrelated LDA (ULDA) [87] together with the kernelized variation on this technique known as KUDA [88].

$$P(\boldsymbol{x} \mid y_i) \approx \frac{1}{(2\pi)^{m/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left(\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_{y_i})\boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_{y_i})\right)$$

Equation 6.25: This equation demonstrates the multivariate Gaussian probability density function for estimating the likelihood of the vector sample *x* in a *m*-dimensional feature space, given the class labelled y_i with a vector class mean μ_{yi} estimated using training data. Under LDA, a common covariance matrix Σ is estimated using training data across all classes; this differs from Quadratic Discriminant Analysis (QDA) [89], which replaces Σ with class-dependent covariance matrices ($\boldsymbol{\Sigma_i}$).

$$P(y_i \mid \boldsymbol{x}) = \frac{P(\boldsymbol{x} \mid y_i)P(y_i)}{P(\boldsymbol{x})} = \frac{P(\boldsymbol{x} \mid y_i)P(y_i)}{\sum_{j=1}^{k} P(\boldsymbol{x} \mid y_j)P(y_j)}$$

Equation 6.26: Using Bayes rule the probability estimated in Equation 6.25 can be used to derive the probability that the class y_i corresponds to the given sample *x*; doing so requires an a priori estimate of the likelihood that a sample of class y_i would appear as an input ($\boldsymbol{P(y_i)}$) for all *k* classes.

When examining the LDA class conditional probability definition in equation 6.25 it is apparent that the single discriminating factor in LDA is the squared Mahalanobis distance which forms its exponent term. When posterior probabilities are not required and the a priori likelihood of each class is considered to be equal, classification can be accomplished simply by using a maximum likelihood estimator of this distance [87]. Yet, a consequence of this dependency on a single feature space-spanning distance metric

is the erosion of the LDA classifier's discriminative ability when classification is performed on feature spaces with a significant number of weakly discriminant features; in this case the weak features would mask the stronger features, resulting in decreased classification performance. The reduced form of LDA mitigates this problem by reducing the dimensionality of the feature space using the supervised dimensionality reduction technique proposed by Fisher [90]. This technique, which is often also referred to as LDA or Fisher Discriminant Analysis (FDA) when used for feature extraction [91], is similar to PCA (section 4.2 of chapter 4) in that it uses Eigen Decomposition to derive a dimensionality reducing transformation matrix, but differs in metric around which the dimensionality reduction is optimized. Under PCA, the zeroed covariance matrix is taken as the optimization criterion and the dimensionality reducing transformation matrix is acquired by performing Eigen Decomposition on this criterion and extracting the subset of the eigenvectors corresponding to the largest eigenvalues returned by the decomposition; the result of this is a dimensionality reduction that preserves as much variance as possible. In contrast, the dimensionality reduction for reduced LDA is based around an optimization criterion known as the Fisher criterion (equation 6.28). Solving for this criterion allows for a reduction in dimensionality that maximizes the separation of class means (the between class scatter matrix $\boldsymbol{S_b}$) while simultaneously minimizing the degree of variance across all classes (the within class scatter matrix $\boldsymbol{S_w}$), thus producing a dimensionality reduction that preserves as much of a separation between each class's Gaussian norms as possible. With a bit of work this criterion can be reformulated as a problem that can be solved via Eigen Decomposition (equation 6.29), having the transformation matrix $\boldsymbol{G}$ defined as the row appended eigenvectors corresponding to the

non-zero eigenvalues. Graphically, the discriminative properties provided by the Fisher technique over PCA become particularly apparent when the dataset variance runs perpendicular to the separation of classes, as demonstrated in figure 6.10.

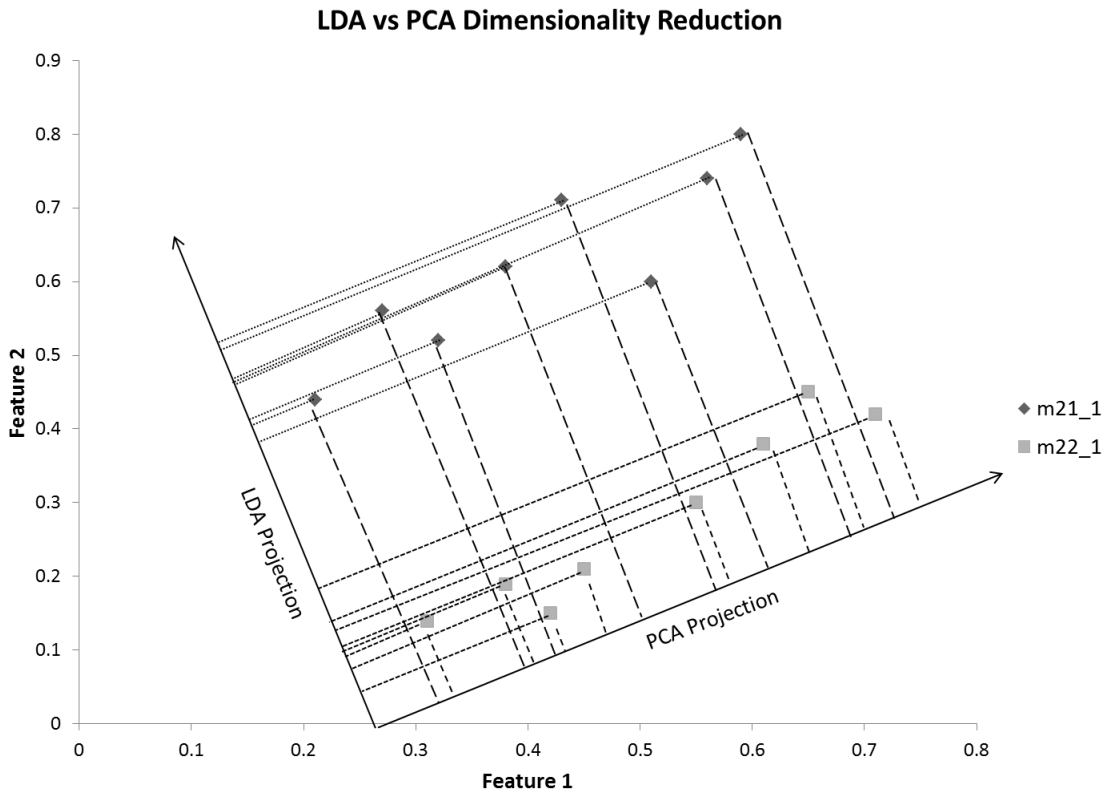

Figure 6.10: This figure compares the dimensionality reduction in PCA, which projects data into the dimension of highest variance, with the reduced LDA dimensionality reduction, which projects data into the dimension maximizing the distance between class distributions. In this case the projection into the LDA dimension produces a result that has the two classes separated, while the projection into the PCA dimension has the data from the two classes interspersed.

$$\boldsymbol{c} = \frac{1}{n}\boldsymbol{A}\boldsymbol{e},\ \boldsymbol{e} = (1,1,\dots,1)^T \in \boldsymbol{R}^n \quad \boldsymbol{c}^{(i)} = \frac{1}{n_i}\boldsymbol{A_i}\boldsymbol{e}^{(i)},\quad \boldsymbol{e}^{(i)} = (1,1,\dots 1)^T \in \boldsymbol{R}^{n_i}$$

$$\boldsymbol{H_w} = \frac{1}{\sqrt{n}}\left[\boldsymbol{A_1} - \boldsymbol{c}^{(1)}\left(\boldsymbol{e}^{(1)}\right)^T, \dots, \boldsymbol{A_k} - \boldsymbol{c}^{(k)}\left(\boldsymbol{e}^{(k)}\right)^T\right]$$

$$\boldsymbol{H_b} = \frac{1}{\sqrt{n}}\left[\sqrt{n_1}\left(\boldsymbol{c}^{(1)} - \boldsymbol{c}\right), \dots, \sqrt{n_k}\left(\boldsymbol{c}^{(k)} - \boldsymbol{c}\right)\right]$$

$$\boldsymbol{S_w} = \boldsymbol{H_w}\boldsymbol{H_w}^T,\ \boldsymbol{S_b} = \boldsymbol{H_b}\boldsymbol{H_b}^T$$

Equation 6.27: This equation demonstrates the definition of the pooled within (S_w) and between (S_b) class scatter matrices that are essential to performing reduced LDA. As demonstrated here they can be represented as the square of two other matrices (their half forms), with *A* being the set of samples, A_i being the set samples associated with class *i*, *c* containing the mean vector for each feature across all *k* classes, $c^{(i)}$ containing the mean vector across class *i*, n being the total number of samples, and n_i being the total number of samples in class *i*.

$$\boldsymbol{W} = \operatorname{argmax}_{\boldsymbol{W}} tr\left(\frac{\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}}{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}}\right)$$

Equation 6.28: This equation demonstrates Fisher's Optimization Criterion, which is based around finding the matrix *W* that maximizes the between class scatter matrix while simultaneously minimizing the within class scatter matrix.

$$\frac{d}{d\boldsymbol{W}}\frac{\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}}{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}} = \frac{\frac{d}{d\boldsymbol{W}}(\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W})\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W} - \frac{d}{d\boldsymbol{W}}(\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W})\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}}{(\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W})^2}$$

$$\Rightarrow \operatorname*{argmax}_{\boldsymbol{W}} tr\left(\frac{\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}}{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}}\right) = \frac{(2\boldsymbol{S_b}\boldsymbol{W})\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W} - (2\boldsymbol{S_w}\boldsymbol{W})\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}}{(\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W})^2} = 0$$

$$\Rightarrow \boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}(\boldsymbol{S_w}\boldsymbol{W}) - \boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}(\boldsymbol{S_w}\boldsymbol{W}) = 0$$

$$\Rightarrow \frac{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}(\boldsymbol{S_b}\boldsymbol{W})}{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}} - \frac{\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}(\boldsymbol{S_w}\boldsymbol{W})}{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}} = \boldsymbol{S_b}\boldsymbol{W} - \frac{\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}}{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}}(\boldsymbol{S_w}\boldsymbol{W}) = 0$$

$$\Rightarrow \boldsymbol{S_b}\boldsymbol{W} = \boldsymbol{\Lambda}(\boldsymbol{S_w}\boldsymbol{W}) \ \ or \ \ \boldsymbol{S_w^{-1}}\boldsymbol{S_b}\boldsymbol{W} = \boldsymbol{\Lambda}\boldsymbol{W}, \quad where \quad \boldsymbol{\Lambda} = \frac{\boldsymbol{W}^T\boldsymbol{S_b}\boldsymbol{W}}{\boldsymbol{W}^T\boldsymbol{S_w}\boldsymbol{W}} \quad and \quad \boldsymbol{G} = \boldsymbol{W_q}$$

Equation 6.29: To solve for the value of *W* that optimizes equation 6.28, the problem can be reformulated into one that is solvable via Eigen Decomposition. In this case *W* will be

formed by the eigenvectors associated with the q non-zero eigenvalues in the diagonal eigenvalue matrix Λ.

Unlike PCA, which, prior to reduction, will produce as many dimensions as there are dimensions of variance in a dataset, the Fisher dimensionality reduction technique is bounded by the number of classes identified in its training dataset. This owes to the fact that a dataset with C unique classes will result in a matrix $\boldsymbol{S_b}$ having a rank with an upper bounding of $C - 1$, and therefore the Eigen Decomposition of equation 6.29 can only ever produce at most $C - 1$ non-zero eigenvalues [87]. Moreover, in datasets containing fewer dimensions than classes this bound on the number of dimensions returned would instead be equal to the number of dimensions in the dataset, thus the upper bound on the number of dimensions produced for reduced LDA can be represented as the minimum of the two possible boundaries (equation 6.30). The lower boundary, in contrast, has no such restrictions and, in using this technique, any dataset could be reduced down to its single most discriminant dimension; however, for the purpose of our research the reduced LDA classifier is always assumed to use a dimensionality reduction equivalent to the upper bound shown equation 6.30 (note that for datasets with a large number of classes this reduction may not be small enough). With a chosen dimensionality reduction strategy in place, once the dimensionality reducing transformation matrix ($\boldsymbol{G}$) has been found, the class conditional probability density function for reduced LDA classifier will take the form shown in equation 6.31.

$$m = \max(p, C-1) \quad with \quad \boldsymbol{x} \epsilon \boldsymbol{R}^p \quad and \quad \boldsymbol{G}^T \boldsymbol{x} \in \boldsymbol{R}^m$$

Equation 6.30: This equation demonstrates the upper bound for dimensionality reduction via the Fisher dimensionality reduction technique over a dataset with *p* dimensions and *C* different classes. In this equation, *x* represents a sample in the dataset while $\boldsymbol{G}^T \boldsymbol{x}$ represents the transformation of *x* into the reduced dimensional space.

$$\tilde{\boldsymbol{x}} = \boldsymbol{G}^T \boldsymbol{x}, \quad \tilde{\boldsymbol{\mu}}_{\boldsymbol{y_i}} = \boldsymbol{G}^T \boldsymbol{\mu}_{\boldsymbol{y_i}}, \quad \tilde{\boldsymbol{S}}_{\boldsymbol{w}} = \boldsymbol{G}^T \boldsymbol{S}_{\boldsymbol{w}} \boldsymbol{G}$$

$$P(\boldsymbol{x} \mid y_i) \approx \frac{1}{(2\pi)^{m/2} \left|\tilde{\boldsymbol{S}}_{\boldsymbol{w}}\right|^{1/2}} \exp\left(\frac{1}{2}\left(\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{\mu}}_{\boldsymbol{y_i}}\right) \tilde{\boldsymbol{S}}_{\boldsymbol{w}}^{-\mathbf{1}} \left(\tilde{\boldsymbol{x}} - \tilde{\boldsymbol{\mu}}_{\boldsymbol{y_i}}\right)\right)$$

Equation 6.31: This equation demonstrates how the class conditional probability density function can be adapted to be used with reduced LDA. To accomplish this, each dataset-dependent input is multiplied with the transformation matrix *G*.

One unfortunate weakness in the above formulation for reduced LDA is the strict requirement that the within-class scatter matrix be invertible. The introduction of a non-invertible singular within-class scatter matrix is a common problem for datasets that contain more samples than dimensions (the SSS problem). To get around this limitation a number of techniques have been proposed including Regularized LDA [92], Nullspace LDA [93], and Penalized LDA [94] among others. Of the proposed solutions the ULDA technique described by Ye in [87] provides a convenient generalization to reduced LDA for situations with both singular and non-singular within-class scatter matrices. Under the ULDA technique the inverted term in the optimization criterion is reformulated using the pseudo-inverse, giving ULDA the valuable property of being equivalent to the standard reduced LDA technique when dealing with the case of a non-singular scatter matrix in the criterion's denominator (in this case the pseudo inverse is equal to the actual inverse). Moreover, when dealing with the case of a singular scatter matrix ULDA acts as an

extension on reduced LDA, using the pseudo-inverse's optimal inverse solution in the case for which an inverse could not be found using standard matrix inversion [95]. The ULDA technique also takes advantage of the fact that the Fisher optimization criterion can be solved using an equivalent form that has the total scatter matrix (equation 6.32) in the place of the within-class scatter matrix [96]. With these adaptations to the standard reduced LDA, the ULDA optimization criterion takes on the new form shown in equation 6.33.

$$\boldsymbol{H}_t = \frac{1}{\sqrt{n}}(\boldsymbol{A} - \boldsymbol{ce}^T), \quad \boldsymbol{S}_t = \boldsymbol{H_t H_t}^T$$

Equation 6.32: This equation demonstrates the derivation of the total scatter matrix, with *A* being the dataset, *c* being the vector of dataset means, and *e* being the vector of 1s defined in equation 6.27.

$$\boldsymbol{G} = arg\max_{\boldsymbol{G}} tr\left((\boldsymbol{G}^T\boldsymbol{S_t}\boldsymbol{G})^+(\boldsymbol{G}^T\boldsymbol{S_b}\boldsymbol{G})\right)$$

Equation 6.33: This equation demonstrates the ULDA optimization criterion. Note that, unlike the Fisher Optimization Criterion, this criterion has the total scatter matrix in the place of the within-class scatter matrix, and uses the pseudo-inverse in the place of the inverse.

The efficient ULDA algorithm that Ye [87] proposed is premised on solving for the matrix $\boldsymbol{X}$ that simultaneously diagonalizes the three different scatter matrices ($\boldsymbol{S_b}$, $\boldsymbol{S_w}$, and $\boldsymbol{S_t}$). Ye was able to show that this could be done using only the half-between class scatter matrix ($\boldsymbol{H_b}$) and half-total class scatter matrix ($\boldsymbol{H_t}$); in Ye's algorithm these half scatter matrices are assumed to have rows corresponding to dimensions and columns corresponding to individual data samples, the transpose of the form these matrices

normally take. The first step in the algorithm involves using SVD to decompose the half total scatter matrix into its orthogonal and diagonal components (equation 6.34); when accounting for the relationship between the scatter matrices it was shown that a large part of the SVD can be ignored and this step could simply be accomplished using the more efficient reduced (or economy) SVD technique [97] (equation 6.35). Having solved for the reduced SVD of $\boldsymbol{H_t}$, it was demonstrated that the relationship between the diagonalized total scatter matrix and the other two scatter matrices could be redefined as shown in equation 6.36. The next step of the algorithm involves a second SVD, this time the full SVD is computed on half of equation 6.36's between-class component, allowing for the derivation of the between-class scatter matrix diagonalization shown in equation 6.37. As a final step in the simultaneous diagonalization of scatter matrices, having discovered the diagonalization of the total and between-class scatter matrices, the orthogonal matrix $\boldsymbol{P}$ discovered in equation 6.37 can be multiplied with equation 6.36 to reveal the diagonalization of all three scatter matrices (equation 6.38); this results in the diagonalization matrix $\boldsymbol{X}$ demonstrated in equation 6.39.

$$\boldsymbol{H_t} = \boldsymbol{U\Sigma V}^T, \quad with \;\; \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma_t} & 0 \\ 0 & 0 \end{pmatrix}, \;\; \boldsymbol{\Sigma_t} \in \boldsymbol{R}^{t \times t}, and\ t = rank(\boldsymbol{S_t}),$$

$$\boldsymbol{H_t} \in \boldsymbol{R}^{m \times n}, \; \boldsymbol{U} \in \boldsymbol{R}^{m \times m}, \; \boldsymbol{\Sigma} \in \boldsymbol{R}^{m \times n}, \; \boldsymbol{V} \in \boldsymbol{R}^{n \times n}$$

Equation 6.34: The full SVD of H_t, shown above, decomposes it into two orthogonal components *U* and *V* as well as a diagonal component Σ. Here the half scatter matrices defined previously have been transposed so that each row represents one of the *m* dimensions in the feature space and each column represents one of the *n* samples in the training dataset.

$$\boldsymbol{S_t} = \boldsymbol{U\Sigma V^T V\Sigma^T U^T} = \boldsymbol{U\Sigma\Sigma^T U^T} = \begin{pmatrix} \boldsymbol{\Sigma_t^2} & 0 \\ 0 & 0 \end{pmatrix},\ \boldsymbol{U} = (\boldsymbol{U_1}, \boldsymbol{U_2}),$$

$$\boldsymbol{U_1} \in \boldsymbol{R}^{m\times t}\ and\ \boldsymbol{U_2} \in \boldsymbol{R}^{m\times(m-t)}$$

$$\begin{pmatrix} \boldsymbol{\Sigma_t^2} & 0 \\ 0 & 0 \end{pmatrix} = \boldsymbol{U^T S_t U} = \boldsymbol{U^T(S_b + S_w)U} = \begin{pmatrix} \boldsymbol{U_1}^T \\ \boldsymbol{U_2}^T \end{pmatrix} \boldsymbol{S_b}(\boldsymbol{U_1},\ \boldsymbol{U_2}) + \begin{pmatrix} \boldsymbol{U_1}^T \\ \boldsymbol{U_2}^T \end{pmatrix} \boldsymbol{S_w}(\boldsymbol{U_1},\ \boldsymbol{U_2})$$

$$= \begin{pmatrix} \boldsymbol{U_1}^T\boldsymbol{S_b U_1} & \boldsymbol{U_1}^T\boldsymbol{S_b U_2} \\ \boldsymbol{U_2}^T\boldsymbol{S_b U_1} & \boldsymbol{U_2}^T\boldsymbol{S_b U_2} \end{pmatrix} + \begin{pmatrix} \boldsymbol{U_1}^T\boldsymbol{S_w U_1} & \boldsymbol{U_1}^T\boldsymbol{S_w U_2} \\ \boldsymbol{U_2}^T\boldsymbol{S_w U_1} & \boldsymbol{U_2}^T\boldsymbol{S_w U_2} \end{pmatrix}$$

$$=>\ \boldsymbol{U^T S_b U} = \begin{pmatrix} \boldsymbol{U_1}^T\boldsymbol{S_b U_1} & 0 \\ 0 & 0 \end{pmatrix}\ and\ \boldsymbol{U^T S_w U} = \begin{pmatrix} \boldsymbol{U_1}^T\boldsymbol{S_w U_1} & 0 \\ 0 & 0 \end{pmatrix}$$

Equation 6.35: This equation demonstrates that reduced SVD, which involves only calculating the first *t* columns of *U*, can be used in the place of full SVD in modeling the relationship between the scatter matrices. In this equation *t* continues to represent the rank of the total scatter matrix and fact that all matrices involved are positive semi-definite mean only the sum of $\boldsymbol{U_1}^T\boldsymbol{S_b U_1}$ and $\boldsymbol{U_1}^T\boldsymbol{S_w U_1}$ will be non-zero with a sum of $\boldsymbol{\Sigma_t^2}$.

$$\boldsymbol{\Sigma_t^2} = \boldsymbol{U_1}^T\boldsymbol{S_b U_1} + \boldsymbol{U_1}^T\boldsymbol{S_w U_1}\ =>\ \boldsymbol{I_t} = \boldsymbol{\Sigma_t^{-1}U_1}^T\boldsymbol{S_b U_1\Sigma_t^{-1}} + \boldsymbol{\Sigma_t^{-1}U_1}^T\boldsymbol{S_w U_1\Sigma_t^{-1}}$$

Equation 6.36: This equation demonstrates how the relationship among scatter matrices, derived in equation 6.35, can be altered to get an identity matrix on one side of the equation.

$$\boldsymbol{B} = \boldsymbol{\Sigma_t^{-1}U_1}^T\boldsymbol{H_b},\ \ with\ SVD\ \ \boldsymbol{B} = \boldsymbol{P\widetilde{\Sigma}Q}^T\ =>\ \boldsymbol{\Sigma_t^{-1}U_1}^T\boldsymbol{H_b} = \boldsymbol{BB^T}$$

$$= \boldsymbol{P\widetilde{\Sigma}Q^T Q\widetilde{\Sigma}\,P^T} = \boldsymbol{P\widetilde{\Sigma}}^2\boldsymbol{P^T} = \boldsymbol{P\Sigma_b P^T}$$

Equation 6.37: In this equation the full SVD is taken over the value given to *B*. By substituting the result back in the between class component of equation 6.36 it can be shown that the orthogonal component *Q* disappears as its product simply becomes an identity matrix, while the diagonal matrix becomes the square of itself.

$$\boldsymbol{P^T I_t P = \Sigma_b + P^T \Sigma_t^{-1} {U_1}^T S_w U_1 \Sigma_t^{-1} P \Rightarrow I_t = \Sigma_b + P^T \Sigma_t^{-1} {U_1}^T S_w U_1 \Sigma_t^{-1} P}$$

$$\boldsymbol{\Rightarrow P^T \Sigma_t^{-1} {U_1}^T S_w U_1 \Sigma_t^{-1} P = I_t - \Sigma_b = \Sigma_w}$$

Equation 6.38: By multiplying the relationship discovered in equation 6.36 with the orthogonal matrix *P*, the relationship between the scatter matrices can be reformulated as a relationship between diagonal matrices.

$$\boldsymbol{X^T S_b X = \begin{pmatrix} \Sigma_b & 0 \\ 0 & 0 \end{pmatrix} \equiv D_b, \quad X^T S_w X = \begin{pmatrix} \Sigma_w & 0 \\ 0 & 0 \end{pmatrix} \equiv D_w, \quad X^T S_t X = \begin{pmatrix} \Sigma_t & 0 \\ 0 & 0 \end{pmatrix} \equiv D_t}$$

$$\boldsymbol{X = U \begin{pmatrix} \Sigma_t^{-1} P & 0 \\ 0 & I \end{pmatrix}}$$

Equation 6.39: Using the findings from equations 6.34 through 6.38, this equation demonstrates the definition of the diagonalization matrix *X* that simultaneously diagonalizes the three scatter matrices.

Finding the solution for the ULDA transformation matrix (***G***) can be simplified when the diagonalization matrix (***X***) is available. In this case the optimization criterion from equation 6.33 can be reduced to the form shown in equation 6.40, by first substituting the scatter matrices for their diagonalized forms, and then simplifying using the cyclic matrix trace property together with the pseudo-inverse properties of equality. Under this new criterion the transformation matrix can be seen as being formed by the product of ***X*** with some unknown matrix ($\widetilde{\boldsymbol{G}}$). In [87] Ye presented a theorem which demonstrated how the maximization of this criterion need only depend on a truncated form of the diagonalization matrix, namely the sub matrix formed by first *q* columns in ***X*** (the columns representing the eigenvectors corresponding to $\boldsymbol{S_b}$'s non-zero eigenvalues). In applying this theorem, it was shown that $\widetilde{\boldsymbol{G}}$ could be decomposed into two different components of which only one would have any effect on the maximization of the

criterion, the $t \times q$ dimensional matrix $\boldsymbol{G_1}$ (for the t rank total scatter matrix), and that this component could be decomposed further to a non-singular $q \times q$ matrix $\boldsymbol{M}$ via removing rows found to have no influence over the criterion. Consequently $\boldsymbol{M}$ could be used in the place of $\widetilde{\boldsymbol{G}}$ as the unknown criterion-influencing component, and, when combined with the truncated diagonalization matrix $\boldsymbol{X_q}$, Ye's theorem proved that the resulting generalized transformation matrix (equation 6.41) would be guaranteed to maximize the criterion given any non-singular value of $\boldsymbol{M}$. In the specific case of ULDA the value of $\boldsymbol{M}$ is set to the identity matrix ($\boldsymbol{I_q}$) so the transformation matrix becomes equivalent to $\boldsymbol{X_q}$ (equation 6.42), and the reduced dimensional space produces features that are uncorrelated from one another. Assigning the discovered value of $\boldsymbol{G}$ to the formerly described reduced LDA classifier then gives us our ULDA classifier.

$$\boldsymbol{G^T S_b G} = \boldsymbol{G}^T(\boldsymbol{X^{-1}})^T(\boldsymbol{X^T S_b X})\boldsymbol{X^{-1}G} = \widetilde{\boldsymbol{G}}^T \boldsymbol{D_b} \widetilde{\boldsymbol{G}}$$

$$\boldsymbol{G^T S_t G} = \boldsymbol{G}^T(\boldsymbol{X^{-1}})^T(\boldsymbol{X^T S_t X})\boldsymbol{X^{-1}G} = \widetilde{\boldsymbol{G}}^T \boldsymbol{D_t} \widetilde{\boldsymbol{G}}$$

$$\widetilde{\boldsymbol{G}} = \boldsymbol{X^{-1}G} = \begin{pmatrix} \boldsymbol{G_1} \\ \boldsymbol{G_2} \end{pmatrix} \; with \;\; \boldsymbol{G_1} \in \boldsymbol{R}^{t \times q} \; and \; \boldsymbol{G_2} \in \boldsymbol{R}^{(m-t) \times q} \;\; for \;\; q = rank(\boldsymbol{S_b})$$

$$\Rightarrow \boldsymbol{G} = arg\max_{\boldsymbol{G}} tr\left(\left(\boldsymbol{G_1}^T \boldsymbol{G_1}\right)^+ \left(\boldsymbol{G_1}^T \boldsymbol{\Sigma_b} \boldsymbol{G_1}\right)\right) =$$

$$arg\max_{\boldsymbol{G}} tr\left(\left(\boldsymbol{G_1 G_1}^+\right)^T \boldsymbol{\Sigma_b} \left(\boldsymbol{G_1 G_1}^+\right)\right)$$

Equation 6.40: This equation demonstrates how the diagonalization matrix *X* can be incorporated into the ULDA optimization criterion so the criterion can be represented in a diagonalized form. Note that with *t* equal to the total scatter matrix rank, the component G_2 does not contribute to the optimization.

$$\boldsymbol{G} = \boldsymbol{X}\widetilde{\boldsymbol{G}} = \boldsymbol{X_q M}, \quad \boldsymbol{R}^{q\times q}$$

Equation 6.41: This equation demonstrates a generalized form for the optimal, non-SSS susceptible LDA transformation matrix, with X_q being the first q (as defined in equation 6.40) vectors in diagonalization matrix and M an arbitrary non-singular matrix of q dimensions.

$$\boldsymbol{G} = \boldsymbol{X_q}$$

Equation 6.42: In ULDA the q-dimensional identity matrix is selected as the choice for M from equation 6.41 as shown above.

In [98] Park et al. demonstrated that the relationship linking the scatter matrices to the reduced LDA optimization criterion could be reformulated so to allow for the maximization of the criterion over a higher dimensional kernel space. A follow up on this research was done by Wang et al. [88], who showed that this new kernel-based optimization criterion could be used in the place of the non-kernel-based optimization criterion when performing ULDA for feature extraction; the resulting algorithm was termed KUDA. In its use for classification, the KUDA algorithm can be derived by simply replacing non-kernelized input with equivalent kernelized input (equation 6.43) during the training and classification phases. During the training phase the two half-scatter matrices, $\boldsymbol{H_b}$ and $\boldsymbol{H_t}$, would then be replaced with their kernelized forms, $\boldsymbol{H_{b(\emptyset)}}$ and $\boldsymbol{H_{t(\emptyset)}}$, to solve for the transformation matrix ($\boldsymbol{G}$), while during the classification phase the within-class scatter matrix ($\boldsymbol{S_w}$) and input sample vector ($\boldsymbol{x}$) would be replaced with their respective kernelized equivalents, $\boldsymbol{S_{w(\emptyset)}}$ and $\emptyset(\boldsymbol{x})$, to obtain the class conditional probability density functions. For computational efficiency, the KUDA classifier takes advantage of the kernel trick, a technique we previously discussed in the

formulation of the kernel-SVM classifier (section 6.3). The work done by Park et al. [87] showed that all reduced LDA inputs could be reformulated as a grouping of dot products and these dot products could be replaced by kernel functions, which, in turn, would still produce dot product values, but this time between a projection of the inputs in a higher dimensional kernel space, moreover without actually needing to perform any computationally expensive feature space projections. Consequently, as was previously demonstrated for the SVM classifier, the samples of a dataset, when treated in the kernel space, may become separable where such separation would not otherwise have been possible using the non-kernelized version of the ULDA classifier.

$$\boldsymbol{K}(\boldsymbol{x_1}, \boldsymbol{x_2} \dots, \boldsymbol{x_n}) = \begin{bmatrix} \kappa(\boldsymbol{x_1}, \boldsymbol{x_1}) & \cdots & \kappa(\boldsymbol{x_1}, \boldsymbol{x_n}) \\ \vdots & \ddots & \vdots \\ \kappa(\boldsymbol{x_n}, \boldsymbol{x_1}) & \cdots & \kappa(\boldsymbol{x_n}, \boldsymbol{x_n}) \end{bmatrix}, \ \boldsymbol{x_i} \epsilon \boldsymbol{A}$$

$$\emptyset(\boldsymbol{x}) \ = \ [\kappa(\boldsymbol{x}, \boldsymbol{x_1}), \kappa(\boldsymbol{x}, \boldsymbol{x_2}) \dots \ \kappa(\boldsymbol{x}, \boldsymbol{x_n})], \ \boldsymbol{x_i} \epsilon \boldsymbol{A}$$

$$\boldsymbol{c}_{\emptyset} = \frac{1}{n}\boldsymbol{K}\boldsymbol{e}, \ \boldsymbol{e} = (1,1, \dots, 1)^T \in \boldsymbol{R}^n \quad \boldsymbol{c}_{\emptyset}^{(\boldsymbol{i})} = \frac{1}{n_i}\boldsymbol{K_i}\boldsymbol{e}^{(\boldsymbol{i})}, \quad \boldsymbol{e}^{(\boldsymbol{i})} = (1,1, \dots 1)^T \in \ \boldsymbol{R}^{n_i}$$

$$\boldsymbol{H_{t(\emptyset)}} = \frac{1}{\sqrt{n}}(\boldsymbol{K} - \boldsymbol{c}_{\emptyset}\boldsymbol{e}^T), \quad \boldsymbol{H_{b(\emptyset)}} = \frac{1}{\sqrt{n}}\left[\sqrt{n_1}\left(\boldsymbol{c}_{\emptyset}^{(\boldsymbol{i})} - \boldsymbol{c}_{\emptyset}\right), \dots, \sqrt{n_k}\left(\boldsymbol{c}_{\emptyset}^{(\boldsymbol{i})} - \boldsymbol{c}_{\emptyset}\right)\right]$$

$$\boldsymbol{S_{w(\emptyset)}} = \frac{1}{n}\sum_{i=1}^{k}\sum_{x \in \boldsymbol{K_i}}\left(\boldsymbol{x} - \ \boldsymbol{c}_{\emptyset}^{(\boldsymbol{i})}\right)\left(\boldsymbol{x} - \ \boldsymbol{c}_{\emptyset}^{(\boldsymbol{i})}\right)^T$$

Equation 6.43: This equation demonstrates the derivation of the kernelized inputs required to perform KUDA with $\boldsymbol{\kappa}(\boldsymbol{x}, \boldsymbol{y})$ being the kernel function. The scatter and half scatter matrices can be derived using the same process as was used for the non-kernelized ULDA but with the input data matrix $\boldsymbol{A} \in \boldsymbol{R}^{n \times m}$ replaced by its Gram kernel matrix *K*. The kernel transformation function $\emptyset(\boldsymbol{x})$ projects the sample *x* into the kernel space with respect to the training data, and, to solve for the scatter matrices, a subset of kernelized samples $\boldsymbol{K_i}$ is obtained for each class *i* belonging to the *k* different classes.

For the purpose of our GRF research, we implemented both the ULDA and KUDA variants of the LDA classifier, using a C# port [99] of the popular Jama matrix manipulation package [100] to perform the required matrix decompositions where required. In our implementation, the Jama SVD algorithm was modified so to allow for the calculation of either the full or economy SVD when requested, as opposed the default Jama SVD behaviour that produces only the economy SVD. We also corrected for a deficiency in Jama whereby the SVD algorithm fails when calculated on matrices having more columns than rows. With regards to parameter optimization, the ULDA algorithm can run independent of external configuration parameters, whereas the optimization of the KUDA algorithm depends on the kernel function configuration used to run it. We decided to use the Gaussian RBF kernel (equation 6.22) for our KUDA classifier, leaving us with a single kernel parameter (γ) to optimize, with optimization being accomplished using an exhaustive search of values. Additionally, to remove any potential bias due to variations in feature scale, we rescaled all input data to these LDA-based classifiers to fall within the range of [0, 1]; while, using our knowledge of the potential subject-sample input distribution, the a priori probabilities ($P(y_i)$ in equation 6.26) of all 10 subjects were considered to be equal at 0.1. Finally, unlike the previous classifiers discussed in this chapter, our LDA-based classifiers required no extra work be done to determine posterior probabilities and the classifier output (equation 6.26) was used directly in our EER calculations.

Our initial attempts at running the ULDA and KUDA classifiers against our GRF data achieved relatively poor recognition results. We discovered that these poor results

resulted from two different problems with our chosen implementation. The first of these problems was related to our application of Bayes Rule to calculate posterior probabilities. In our initial implementation we neglected the fact that the distances in the exponent of the unscaled class probabilities may take on very large negative values, some so large that they result in underflow in a typical application. This led to cases where all class probabilities would end up being assigned value of zero and the scaled classifier posterior probability would come out as being undefined. To address this issue we added an additional step prior to applying Bayes Rules. The new step involves first computing the logarithms for each of the class probabilities, then subtracting a constant from each of the logarithms to assign a value of zero to the greatest logarithmic probability, and finally exponentiating the logarithmic terms (see equation 6.44). After applying this step we find that at least one of the unscaled class probabilities will always be assigned a value of one and the classifier's posterior probability output will never be undefined. The second problem we encountered came as the result of our chosen LDA dimensionality reduction technique. What we discovered was that the ULDA-based dimensionality techniques had a strong tendency to overfit our training data. Consequently, our dimensionally-reduced training data class distributions, and, by extension our pooled within class covariance matrix, contained an incredibly small degree of variance, producing class boundaries that perfectly separated training samples by class yet proved poor for classification of testing data samples. In [101] it was found that the ULDA transformation causes the samples of each class to converge to a single point per class under the mild condition (equation 6.45), which typically leads to classifier overfitting. In that paper the regularized-LDA technique known as RLDA was suggested as an overfitting-resistant alternative. For our

research, we altered the handling of our ULDA output to assume a greater degree of variance in the calculation of our class probabilities. Taking into account the fact that the ULDA dimensionality reduction leads to features that are uncorrelated (producing diagonal covariance matrices with the variances of each feature represented down the diagonal) we replaced the transformed within-class scatter matrix of equation 6.31 with a number of different diagonal matrices and tested the classifier performance against them. In our work we found that by simply using an identity matrix in the place of the much smaller transformed within-class scatter matrix, the classifier was better able to generalize class boundaries, providing far better classification results. The new class conditional probability function used in our work is shown in equation 6.46, while the optimization of the kernel parameter for the KUDA algorithm under this new classification approach is demonstrated in figure 6.11.

$$L_{y_i} = \ln\big(p(\boldsymbol{x}|y_i)\big) + \ln\big(p(y_i)\big),\ L_i = L_{y_i} - C_{max_{y_j}},\quad j = 1, \dots k, \qquad P(y_i|\boldsymbol{x}) = \frac{e^{L_i}}{\sum_{j=1}^{k} e^{L_j}}$$

Equation 6.44: This equation demonstrates the logarithm trick we used to avoid numeric underflow in the calculation of our posterior probabilities. In this equation we calculate the logarithms for all our class probabilities (the conditional probabilities multiplied with the a priori class probabilities), then subtract a value equivalent to the maximum logarithm $\boldsymbol{C_{max_{y_j}}}$ for all *k* classes.

$$rank(S_b) + rank(S_w) = rank(S_t)$$

Equation 6.45: This equation demonstrates the mild condition, which occurs when the rank of the sum of the within and between scatter matrices is equal to that of the total scatter matrix.

$$P(\boldsymbol{x} \mid y_i) \approx \frac{1}{(2\pi)^{\frac{m}{2}}|\boldsymbol{E}|^{\frac{1}{2}}} \exp\left(\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_{y_i})\boldsymbol{E}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_{y_i})\right)$$

$$\Rightarrow P(\boldsymbol{x} \mid y_i) \approx \frac{1}{(2\pi)^{\frac{m}{2}}} \exp\left(\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_{y_i})(\boldsymbol{x} - \boldsymbol{\mu}_{y_i})\right), \quad when \quad \boldsymbol{E} = \boldsymbol{I_q}$$

Equation 6.46: In this variant of the class conditional probability definition the diagonal covariance matrix (*E*) is determined based on its ability to generalize class boundaries. In our implementation the identity matrix for this *q*-dimensional space was chosen, simplifying the calculations required.

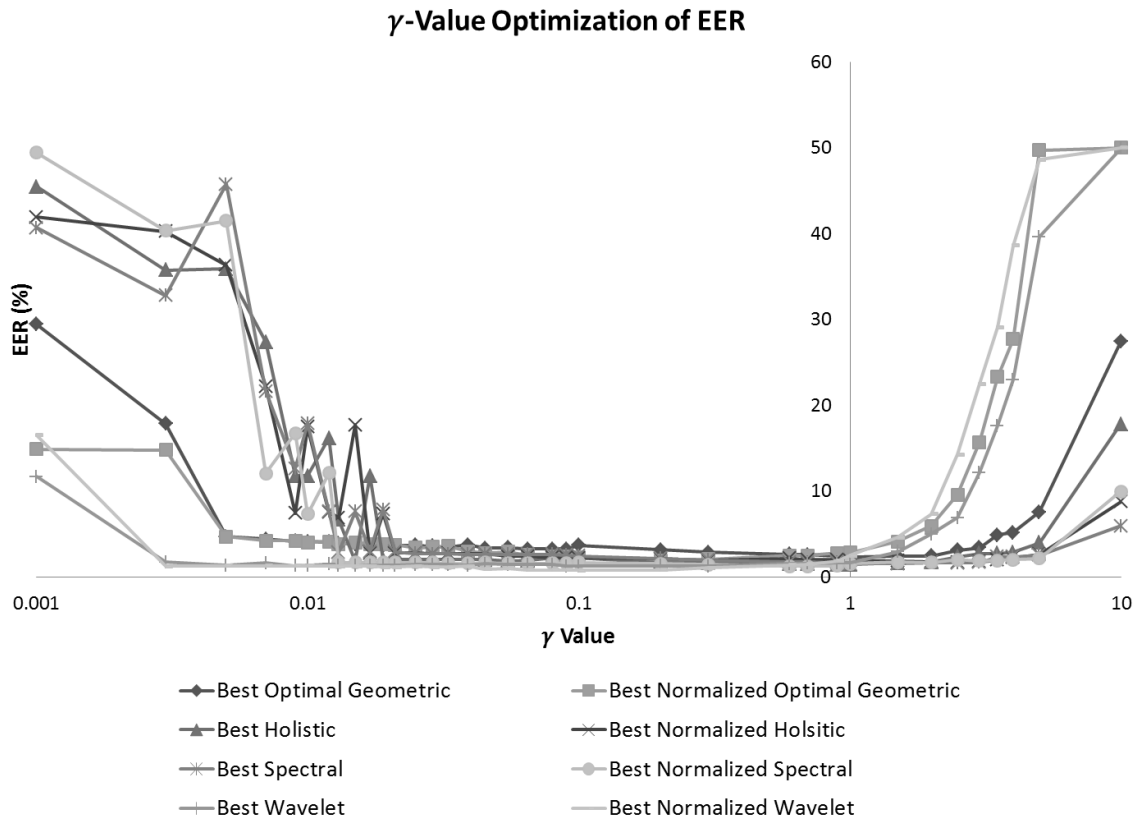

Figure 6.11: This figure demonstrates the impact of the kernel optimized parameter γ on the cross-validated EER calculated across our best performing feature preprocessing techniques.

The optimal GRF recognition results for our implementation of the ULDA and KUDA classifiers over our best previously discovered feature spaces are demonstrated in table

6.4. We found that the kernelized KUDA classifier performed considerably better than the ULDA classifier on the wavelet and geometric feature spaces, but was generally not as strong at classifying the holistic and spectral feature spaces. Moreover, we found that these LDA classifier variants generally performed slightly worse than SVM classifier over the holistic and spectral feature spaces, yet performed noticeably better than any other in the wavelet feature spaces. Also, as was the case in our previous SVM and MLP classifiers, we found that our LDA variances performed worse than KNN in the geometric feature spaces, but, again, this may be explained by the training bias toward KNN. Our findings paralleled those of [7] in that our LDA classifier achieved similar results to those of the SVM classifier; however, little detail was given in [7] regarding the LDA variant used, and, to our knowledge, we are the first to use ULDA and KUDA for GRF recognition. Additionally, it must be noted that these classification results were idealized in that real a priori probabilities were actually known during computations, something that is often not the case in other classification scenarios. Nevertheless, our findings leave a lot of room for future improvement; for instance, classification performance could be improved by deriving class conditional covariance matrices (equation 6.46) that better reflect the variance of features in the reduced LDA space than our chosen identity matrix, or, alternatively, the powerful dimensionality reduction abilities of these techniques could be re-purposed to discover more discriminative features during feature extraction.

Optimal ULDA Classifier Results

Pre-processor	Threshold	Cross Validated EER (%)	EER Improvement (%)
Best Optimal Geometric	0.0468	7.12222	-434.1
Best Normalized Optimal Geometric	0.0812	5.45555	-2968.8
Best Holistic	0.1937	1.5	41.3
Best Normalized Holistic	0.2546	1.5	26.6
Best Spectral	0.3046	1.21111	40.1
Best Normalized Spectral	0.2796	1.16666	36.7
Best Wavelet	0.0625	3.81111	-195.6
Best Normalized Wavelet	0.0812	2.65555	-141.4

Optimal KUDA Classifier Results

Pre-processor	γ	Threshold	Cross Validated EER (%)	EER Improvement (%)
Best Optimal Geometric	1	0.1171	2.3	-72.5
Best Normalized Optimal Geometric	0.2	0.1734	1.87777	-956.2
Best Holistic	0.9	0.1671	1.45555	43
Best Normalized Holistic	2.5	0.1531	1.75555	14.1
Best Spectral	1.5	0.2015	1.52222	24.7
Best Normalized Spectral	0.6	0.1656	1.26666	31.3
Best Wavelet	0.09	0.3156	1.24444	3.4
Best Normalized Wavelet	0.1	289	0.78888	28.2

Table 6.4: These tables demonstrate the best performance achieved by the ULDA and KUDA classifiers for each preprocessing technique. The threshold shown is the posterior-probability threshold at which the EER improvement was calculated and the EER improvement represents the improvement in recognition performance achieved by the optimal LDA variant over the results calculated in the previous two chapters.

6.5 Least Squares Probabilistic Classifier

The results achieved by the LDA classifier described in the previous section demonstrated that generative classifiers can in fact be effective for the purpose of GRF recognition. The LDA classifier, however, has several drawbacks, including the fact that the previously discussed LDA algorithms required the use of intensive computations, such as SVD, on matrices that grow with the size of the training dataset, as can be seen from the derivation of the half total scatter matrix in equation 6.34. This problem could potentially be mitigated by using LDA as a binary classifier and performing classification through the one-against-one strategy previously used with the SVM classifier, but this would take away from the convenience of having a classifier that directly generates single dataset-wide probabilistic output values and might reduce recognition performance. An alternative classification technique proposed by Sugiyama in [42] was designed to directly model posterior probability, but in a manner that would not require the performing of computations on any matrix with a dimensionality larger than the largest subset of class samples in the training data. This efficient algorithm, referred to as Least Squares Probabilistic Classification (LSPC), opts to solve for optimal posterior probability models via the least squares learning technique, as opposed to relying on the multivariate Gaussian modeling that forms the core of the posterior probability derivation in the LDA classification approach. To our knowledge the LSPC classification technique has never before been used for the purpose of GRF recognition, however, in his paper Sugiyama demonstrated that the algorithm could achieve strong classification performance and fast training times for complicated datasets including handwritten digits and satellite imagery.

The LSPC classifier is a discriminative eager learning-based classification algorithm that generates its probabilistic models by first deriving the class probability search spaces as a parameterized linear combination of training data-based basis functions, and then solves for this system of linear equations to discover the probability models that most closely reflect the true posterior class probability spaces. In LSPC the individual parameterized class models take the form demonstrated in equation 6.47, where the parameter $\boldsymbol{\alpha}$ is optimized to correspond with the value it would take for the optimal probability representation of the training data. Having established the parameterized posterior class probability models, the LSPC optimization criterion, shown in equation 6.48, can be represented as the mean squared error between these parameterized probability models and the true probability values, which are established by the analysis of the training data as per equation 6.49. The resulting equation is convex and thus finding its global minimum, which occurs when its derivative is equal to zero, will reveal the value of $\boldsymbol{\alpha}$ corresponding to the optimal probability model (equation 6.50). To avoid the potential for model overfitting the solution to $\boldsymbol{\alpha}$ contains an additional L2 regularization term [102], where the regularization input parameter λ is a scalar value that must be determined prior to training.

$$q(y|\boldsymbol{x};\ \boldsymbol{\alpha}) = \sum_{l=1}^{b} \alpha_l \emptyset_l(\boldsymbol{x}, y) = \boldsymbol{\alpha}^T \emptyset(\boldsymbol{x}, y), \quad where \quad \emptyset(\boldsymbol{x}, y) \geq \mathbf{0_b},\ \forall(\boldsymbol{x}, y)$$

Equation 6.47: This equation demonstrates the probability model parameterized by α, which reflects the likelihood that a given sample x belongs to a given class y. This model is composed of a series of b basis functions ϕ corresponding to a sequence of sample pairs, each producing a value greater than zero.

$$J(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{y=1}^{c}\int\left(q(y|\boldsymbol{x};\ \boldsymbol{\alpha}) - p(y|\boldsymbol{x})\right)^2 p(\boldsymbol{x})\,d\boldsymbol{x}$$

$$= \frac{1}{2}\sum_{y=1}^{c}\int q(y|\boldsymbol{x};\ \boldsymbol{\alpha})^2 p(\boldsymbol{x})d\boldsymbol{x} - \frac{1}{2}\sum_{y=1}^{c}\int q(y|\boldsymbol{x};\ \boldsymbol{\alpha})p(\boldsymbol{x},y)d\boldsymbol{x} + Const$$

$$= \frac{1}{2}\boldsymbol{\alpha}^2\boldsymbol{H}\boldsymbol{\alpha} - \boldsymbol{h}^T\boldsymbol{\alpha} + Const$$

Equation 6.48: This equation presents the optimization criterion for the LSPC classifier. As shown above, the optimization criterion is represented as the mean squared error between the true probability *p(y|x)* and modeled probability *q(y|x; α)* across samples from all training classes (*c*).

$$\boldsymbol{H} = \sum_{y=1}^{c}\int \emptyset(\boldsymbol{x},y)\emptyset(\boldsymbol{x},y)^T p(\boldsymbol{x})d\boldsymbol{x} \ \approx\ \widehat{\boldsymbol{H}} = \frac{1}{n}\sum_{y=1}^{c}\sum_{i=1}^{n}\emptyset(\boldsymbol{x},y)\emptyset(\boldsymbol{x},y)^T,$$

$$\boldsymbol{h} = \sum_{y=1}^{c}\int \emptyset(\boldsymbol{x},y)p(\boldsymbol{x},y)d\boldsymbol{x} \ \approx\ \widehat{\boldsymbol{h}} = \frac{1}{n}\sum_{i=1}^{n}\emptyset(\boldsymbol{x_i},y_i)$$

Equation 6.49: This equation demonstrates how the unknown probability densities components that form the values for *H* and *h* in optimization criterion can be estimated by computing the sample averages.

$$\widehat{\boldsymbol{\alpha}} = \operatorname{argmin}_{\boldsymbol{\alpha}\epsilon\boldsymbol{R}^b}\left[\frac{1}{2}\boldsymbol{\alpha}^2\widehat{\boldsymbol{H}}\boldsymbol{\alpha} - \widehat{\boldsymbol{h}}^T\boldsymbol{\alpha} + \boldsymbol{\lambda}\boldsymbol{\alpha}^T\boldsymbol{\alpha}\right]$$

$$=> \left(\widehat{\boldsymbol{H}} + \boldsymbol{\lambda}\boldsymbol{I_b}\right)\boldsymbol{\alpha} = \widehat{\boldsymbol{h}} \ => \widehat{\boldsymbol{\alpha}} = \left(\widehat{\boldsymbol{H}} + \boldsymbol{\lambda}\boldsymbol{I_b}\right)^{-1}\widehat{\boldsymbol{h}}$$

Equation 6.50: This equation shows how the optimization criterion can be reformulated so to solve for the value that the parameter *α* takes when optimized ($\widehat{\alpha}$). Because the criterion is convex, this occurs when its derivative is equal to zero.

In [42] Sugiyama demonstrated that a final posterior class probability could be acquired by taking the aforementioned probability models, rounding any negative outputs to zero, and performing a normalization step; yet he went on to show that the solution could be found more efficiently when appropriate basis functions are chosen. Sugiyama chose to separate the basis input and output parameters (***x*** and *y*) so that each was transformed by

a different kernel function, with the Kronecker delta kernel [103] chosen to handle the output values. Having formed the basis from these two different kernel functions, the parameterized probability models can take the form shown in equation 6.51; or, alternatively, accounting for the effect of the delta function, the models could be computed separately in a class-wise manner as demonstrated in equation 6.52. From here Sugiyama went on to show that further simplification could be accomplished by choosing a localized kernel to handle the basis input parameter; a localized kernel being a kernel whose values are at their greatest nearest known class contributing training points and become smaller as you move further away from those points. In this case the kernels would make the greatest contributions to probability values in regions where kernels overlap and little-to-no contribution in regions with few or no training samples for a given class (see figure 6.12). Consequently, when using a localized kernel, such as the Gaussian kernel (the kernel used in the SVM and LDA classifiers), the class probability models could be reduced to the form shown in equation 6.53. In making this simplification, the maximum dimensionality of the matrix $\boldsymbol{H}$ would be reduced from a square matrix with a dimensionality equivalent to the number of samples in the training dataset (n) to a square matrix with a dimensionality only as large as the number of training samples for the examined class y (n_y); this drastically decreases the computational work needed to solve for the optimal value of $\boldsymbol{\alpha}$, as shown in equation 6.54. With this more efficient approach to computing the probability models, the final posterior class probabilities could be computed using Sugiyama's negative value rounding and normalization approach (equation 6.55).

$$q(y|\boldsymbol{x};\ \boldsymbol{\alpha}) = \sum_{y'=1}^{c}\sum_{l=1}^{n}\alpha_l^{(y')}K(\boldsymbol{x},\boldsymbol{x_l})\delta_{y,y'}, \qquad \delta_{y,y'} = \begin{cases} 1 & if\ y = y' \\ 0 & otherwise \end{cases}$$

Equation 6.51: In the above equation the basis function from equation 6.47 is split into two different kernel functions. With the sample input values (*x*) forming the parameters for some arbitrary kernel function and the sample output class labels (*y*) forming the parameters for the Kronecker delta function. In this case, *n* represents the number of samples in the training dataset and *K* the arbitrary kernel function.

$$q(y|\boldsymbol{x};\ \boldsymbol{\alpha}) = \sum_{l=1}^{n}\alpha_l^{(y)}K(\boldsymbol{x},\boldsymbol{x_l})$$

Equation 6.52: This equation shows the simplified form of the parameterized class probability model that results when applying the values of the delta function from equation 6.51.

$$q(y|\boldsymbol{x};\ \boldsymbol{\alpha}) = \sum_{l=1}^{n_y}\alpha_l^{(y)}K\left(\boldsymbol{x},\boldsymbol{x}_l^{(y)}\right)$$

Equation 6.53: This equation represents a further simplification on the parameterized class probability model that results when a localized kernel is chosen. In this case the kernel function need only be computed over the n_y samples belonging to the examined class *y*.

$$\widehat{H}_{l,l'}^{(y)} = \frac{1}{n}\sum_{i=1}^{n}K\left(\boldsymbol{x_i},\boldsymbol{x}_l^{(y)}\right)K\left(\boldsymbol{x_i},\boldsymbol{x}_{l'}^{(y)}\right), \quad \hat{h}_l^{(y)} = \frac{1}{n}\sum_{i=1}^{n_y}K\left(\boldsymbol{x}_i^{(y)},\boldsymbol{x}_l^{(y)}\right),$$

$$\boldsymbol{\alpha}^{(y)} = \left(\alpha_1^{(y)},\dots,\alpha_{n_y}^{(y)}\right)^T, \quad \boldsymbol{\alpha}^{(y)} = \left(\widehat{\boldsymbol{H}}^{(y)} + \boldsymbol{\lambda}\boldsymbol{I}_{\boldsymbol{n_y}}\right)^{-1}\widehat{\boldsymbol{h}}^{(y)}$$

Equation 6.54: With the simplified parameterized probability model from equation 6.53, the solution to the optimal value of α takes on the value demonstrated above. In this case α^y must be calculated for each class *y*, but now the matrix (*H*) and vector (*h*) required for the optimization will only ever have a dimensionality equal to the number of training samples in the class being optimized.

$$\hat{p}(y|\boldsymbol{x}) = \frac{\max\left(0,\ \sum_{l=1}^{n_y} \hat{\alpha}^{(y)} K\left(\boldsymbol{x}, \boldsymbol{x}_l^{(y)}\right)\right)}{\sum_{y'=1}^{c} \max\left(0,\ \sum_{l=1}^{n_{y'}} \hat{\alpha}^{(y')} K\left(\boldsymbol{x}, \boldsymbol{x}_l^{(y')}\right)\right)}$$

Equation 6.55: Following the example in [42] the parameterized class probability model is normalized by the sum of all other parameterized class probability models to produce the probability estimate. In the case where the parameters α take on negative values and result in a model returning a negative estimate, the returned values are rounded up to zero.

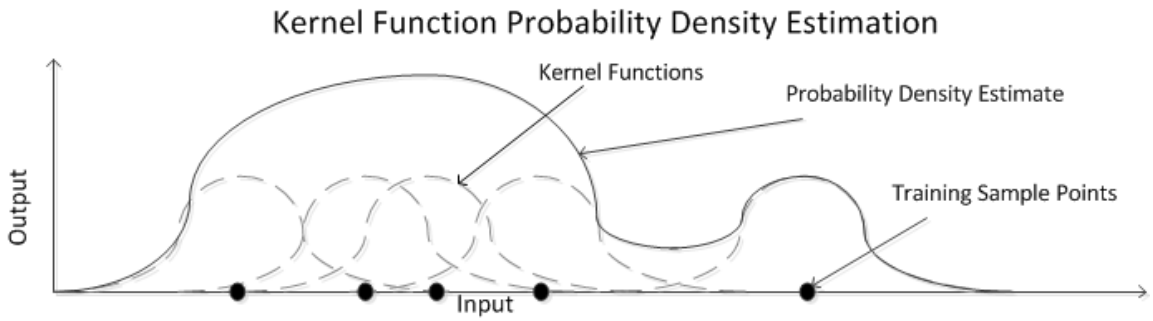

Figure 6.12: This figure demonstrates how localized kernels could be combined to give the highest probability density estimate values in regions with many samples and lower values elsewhere.

For the purpose of our research we initially implemented the efficient variant of the LSPC algorithm described above, making use of the C# port [99] for the Jama matrix manipulation package to perform the $\boldsymbol{\alpha}$ optimization step. Externally, this left us with three configurable inputs to be accounted for when performing classification: the input training feature values, the regularization parameter, and the kernel parameters. To mitigate any potential bias in the feature input values we went with the scaling technique used in our previous four classifiers and rescaled the input for each feature input to fall between the values of 0 and 1 prior to training and classification. The remaining input parameters were assigned by the LSPC classification algorithm to take some pre-determined performance-optimizing values and thus required some tuning to achieve

desirable performance. Moreover, the LSPC classifier was designed such that it could be assigned any arbitrary localized kernel function. In our implementation we used the variant of Gaussian kernel previously used in the implementation by Sugiyama in [42] (see equation 6.66), leaving us with a single kernel parameter $\boldsymbol{\sigma}$. This allowed us to optimize our GRF recognition performance by performing an exhaustive 2-dimensional grid search over the regularization and kernel parameters using an analytically determined set of possible values for each.

$$K(\boldsymbol{x}, \boldsymbol{x}') = exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{2\sigma^2}\right)$$

Equation 6.66: This equation demonstrates the variant of the Gaussian kernel we used to perform LSPC classification.

$$E_{max} = \max_{q(y \mid \boldsymbol{x};\boldsymbol{\alpha})>0,\ \boldsymbol{\alpha}_l^{(y)}>0}\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{x}_l^{(y')}\right\|^2}{2\sigma^2}\right),$$

$$L(y|\boldsymbol{x}) = \frac{1}{e^{Emax}}\frac{\max\left(0,\ \sum_{l=1}^{n_y}\widehat{\alpha}^{(y)}exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{x}_l^{(y)}\right\|^2}{2\sigma^2}\right)\right)}{\sum_{y'}^{c}\max\left(0,\ \sum_{l=1}^{n_{y'}}\widehat{\alpha}^{(y')}exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{x}_l^{(y')}\right\|^2}{2\sigma^2}\right)\right)}$$

$$= \frac{\max\left(0,\ \sum_{l=1}^{n_y}\widehat{\alpha}^{(y)}exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{x}_l^{(y)}\right\|^2}{2\sigma^2} - E_{max}\right)\right)}{\sum_{y'}^{c}\max\left(0,\ \sum_{l=1}^{n_{y'}}\widehat{\alpha}^{(y')}exp\left(-\frac{\left\|\boldsymbol{x}-\boldsymbol{x}_l^{(y')}\right\|^2}{2\sigma^2} - E_{max}\right)\right)}, \qquad \hat{p}(y|\boldsymbol{x}) = \begin{cases} L(y|\boldsymbol{x}) & for\ L(y|\boldsymbol{x}) > 0 \\ 0 & for\ L(y|\boldsymbol{x}) < 0 \end{cases}$$

Equation 6.67: This equation demonstrates the modifications we made to equation 6.65. In this case probability estimates will be assumed to have a value of zero when the sum of the

probabilities found across known classes is equal to zero and exponential terms are adjusted by the value E_{max} to avoid numeric underflow.

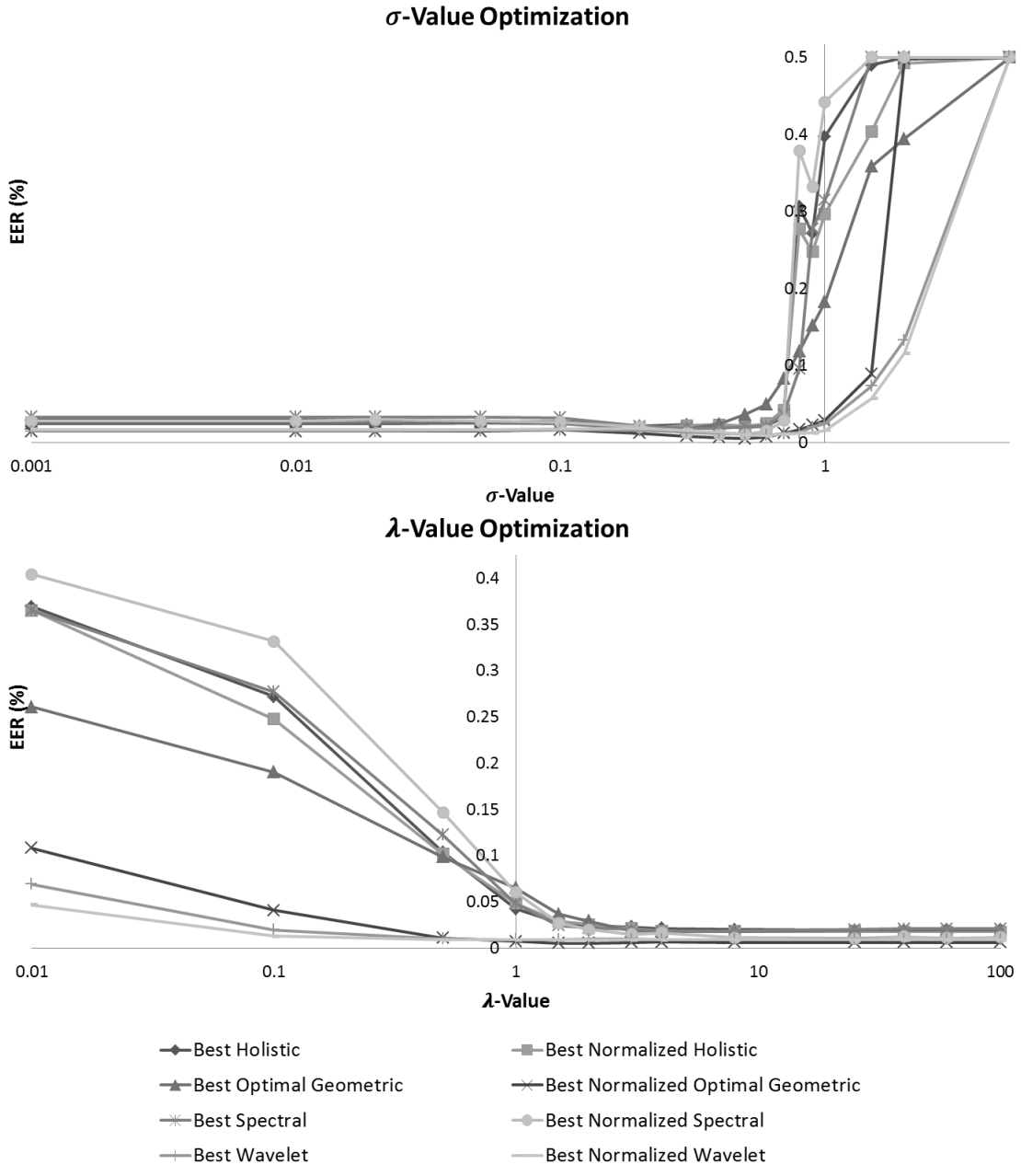

Figure 6.13: This figure compares our best cross-validated EER values achieved with the LSPC classifier to the optimization parameters used to achieve them for all our best performing preprocessing techniques. For either given parameter value, optimization was carried out by testing the given parameter with every possible value for the other parameter and returning the EER for the best combination.

During our initial optimization analysis of the LSPC algorithm we encountered several problems regarding numerical edge cases and computational limitations. First of all, the original LSPC algorithm implicitly assumed that for any given input value at least one of the posterior class probability models will produce a value greater than zero. In practice, using our GRF training data, this assumption proved to be untrue and, as a result, undefined posterior probabilities were encountered. Furthermore, we found that the kernel exponent terms in the LSPC classifier function (equation 6.65) took on very large negative values to the point that we frequently encountered numeric underflow during our GRF recognition tests and were unable to assign non-zero posterior class probabilities. To address these issues we made several modifications to the original algorithm. With regards to the assumption of there being at least one non-zero posterior class probability we first calculated the sum of all class probability model outputs, and in the case of a zero value probability sum we assigned a value of zero for the requested posterior class probability. Thus in the case where only negative model probability outputs were encountered none of the trained GRF subjects (classes) would be accepted as the owner of the given input sample. To avoid the possibility of numeric underflow we used a trick similar to the logarithmic trick for avoiding numeric underflow discussed in section 6.4. In this case we adjusted the algorithm to search for an exponent by which to divide our kernelized samples to give our largest exponent term, corresponding to a positive $\boldsymbol{\alpha}$, a value of 1, while avoiding underflow by applying the law of exponents for merging division terms. To account for the fact that in some cases the largest positive exponential term corresponded with probability models that came out to be negative overall we modified our search for exponent denominator to ignore models with negative outputs.

Our new posterior class probability output function resulting from these changes is demonstrated in equation 6.67, while the best achieved GRF recognition rates for each tested parameter value over our modified LSPC classifier variant are demonstrated in figure 6.13.

After running our results we found that the LSPC parameter optimization curves for our kernel (σ) and regularization (λ) parameters mirrored our observed findings for the performance of equivalent parameter optimizations with the SVM classifier (section 6.3); it can be seen that GRF recognition performance tended to be better for both smaller kernel parameter values (less than 1) and larger regularization parameter values (greater than 1). With regards to overall GRF recognition performance (table 6.5), the cross validated results that the LSPC classifier achieved were better than all other non-KNN classifiers on the geometric feature spaces, while it also consistently performed better than the KNN classifier on all non-geometric feature spaces. Of particular note was the strong recognition performance by the LSPC classifier when applied to the wavelet feature spaces, with the LSPC classifier achieving the best performance of all classifiers in the non-normalized wavelet feature space. This together with its performance for the geometric and LLSRDTW-normalized spectral feature spaces may suggest the LSPC classifier benefitted more than other classifiers when applied to feature spaces that were extracted using greater degrees of supervision during extractor training. Nevertheless, the GRF recognition results achieved by the LSPC classifier in this section have demonstrated that it can be competitive with other classifiers without the large training

dataset size-based performance penalty that would be experienced in many other algorithms that model posterior class probabilities such as the previously examined LDA.

Optimal LSPC Classifier Results

			Cross Validated		
Pre-processor	**λ**	**σ**	**Threshold**	**EER (%)**	**EER Improvement (%)**
Best Optimal Geometric	8	0.3	0.1921	1.8	-35
Best Normalized Optimal Geometric	1.5	0.5	0.314	0.51111	-187.5
Best Holistic	25	0.4	0.1581	1.92222	24.7
Best Normalized Holistic	40	0.2	0.2453	1.77777	13
Best Spectral	4	0.4	0.1822	1.82222	9.8
Best Normalized Spectral	3	0.6	0.1453	1.06666	42.1
Best Wavelet	1	0.6	0.3703	0.94444	26.7
Best Normalized Wavelet	1	0.6	0.3187	0.87777	20.2

Table 6.5: This table demonstrates the best performance achieved by the LSPC classifier for each preprocessing technique. The threshold shown is the threshold at which the EER value was calculated (a value between 0 and 1 derived from the raw posterior probability output) and the EER improvement represents the improvement in recognition performance achieved by the optimal LSPC variant over the results calculated in the previous two chapters.

6.6 Summary

This chapter covered the use of classifiers for the purpose of performing footstep GRF-based person recognition. In the biometric system previously discussed in chapter 3, the classification component is the part responsible for learning patterns from the data and producing an output value that can be used to either accept or reject a data sample as belonging to a data class (in verification mode), or identify the class that most closely matches a sample (in identification mode). For the purpose of our research all results were collected in verification mode and returned in the form of an EER. Consequently, each classifier examined in this chapter was configured to accept a sample and a GRF subject label as input then output a single posterior probability representing the classifier's confidence that the provided sample belonged to the provided subject. Through our introduction to the concept of classification we discussed how classifiers could be categorized as being either instance-based or eager learning-based in addition to following either a discriminative or generative model, with examples of each category of classifier being examined. In total five different powerful classification techniques were studied, four of which were examined in previous GRF-based recognition studies, and one (LSPC) that had never-before been tested for the purpose of GRF-based recognition.

Theoretical background was provided as our chosen classifiers were examined, exposing areas in which each could benefit from the optimization of internal parameters. The best results obtained after performing these optimizations over each classifier, in combination with our top performing preprocessors from chapters 4 and 5, are demonstrated in table 6.6. These results may appear to indicate that the normalized optimal geometric

preprocessor in combination with the KNN classifier performed best; however, as was mentioned in the proceeding sections, the results obtained from our development dataset were strongly biased in favour of the KNN classifier, on account of its use in optimizing the preprocessors. In the next chapter we take the top performing biometric system configurations from our experimental design, those discovered through this and the previous three methodology chapters, and apply them to a set of previously unseen data (our evaluation dataset). Having limited the bias toward any particular classifier or preprocessor, the aim of the next chapter is to gather the results needed to verify our two problem statement assertions and better assess our GRF-recognition performance.

Classifier GRF Recognition Performance

Feature Space	Normalizer	Best Classifier	EER (%)
Optimal Geometric	-	KNN	1.33333
Optimal Geometric	LLSR	KNN	0.17777
Holistic	-	SVM	0.96666
Holistic	L1	SVM	1.38888
Spectral	-	LDA (ULDA)	1.21111
Spectral	LLSRDTW	SVM	0.83333
Wavelet	-	LSPC	0.94444
Wavelet	LTN	LDA (KUDA)	0.78888

Table 6.6: This table compares the best GRF recognition performance achieved across each preprocessor when used in combination with our five chosen classification techniques.

Chapter 7

Measured Performance

In chapter 3 the concept of separating data into an evaluation and development dataset for optimization and testing was discussed. In the previous three chapters we used our development dataset to select the biometric system configurations that performed best when applied for GRF footstep recognition. This chapter puts those biometric configurations to the test by applying them to the previously unseen samples in our evaluation dataset. Using the evaluation dataset, this chapter aims to present a set of results not influenced by any of the biases introduced during the optimization of the development dataset.

In the sections that follow our evaluation dataset results will be presented from three different perspectives. The first section presents a generalized assessment of our GRF recognition performance when applied over our entire evaluation dataset for our best performing classifiers from chapter 6, while the two sections that follow examine the effectiveness of stepping speed-based normalization and influence of shoe type, respectively. It is hoped that the findings obtained in this chapter will help verify our problem statement assertions and produce a more accurate representation of the GRF recognition performance that can be expected from our top biometric system configurations under real-world conditions.

7.1 Evaluation Dataset

In chapter 6 we derived a set of biometric system configurations to achieve optimal GRF recognition performance when computed over our development dataset. Although these configurations demonstrated powerful recognition capabilities, the process of optimization has a tendency to overfit specific characteristics reflecting the dataset on which the optimization was performed. Consequently, the optimal configurations discovered in chapter 6 may be far from optimal when applied in a more general setting to previously unseen real world data. To better gauge the effectiveness of our biometric system configurations, in this section and the sections that follow we re-apply some of our best performing experimental biometric configurations to our evaluation dataset.

Our evaluation dataset, as previously described in chapter 3, is made up of 199 samples from 10 different subjects, divided according to two different shoe types (100 collected with subjects wearing a Verona runner and 99 collected with subjects wearing an Orin runner). Samples were split evenly across shoe types with all but one subject having 10 steps per shoe type; the remaining subject (m22_1) was missing a single Orin sample. For the purpose of our research we opted to use same k-fold cross validation and the EER metric as was used to assess our development dataset performance to assess GRF recognition performance over our evaluation dataset; however, the missing sample introduced some complexity to this approach. To account for the missing step and still use k-fold cross validation we would need to either ignore the training folds that lined up with the missing sample, pull in an additional sample from the testing set to replace the

missing sample when needed, or allow the subject with the missing sample to be trained with one fewer sample than his peers. For the purpose of our research we opted for the last option and once again went with 5 training samples per subject in our tests. Thus tests that included the Orin shoe type for training could be expected to perform slightly worse than those that were purely trained with the Verona shoe type, on account of one subject being trained on only 4 samples in some of the cross validation folds.

When applying the biometric system configurations derived from our development system in the previous chapters we also faced a problem regarding which of the machine learning-based preprocessor transformations (i.e. PCA, WPD, LLSRDTW) to use in our evaluation testing. To this point a different transformation was computed for each cross validation fold over the development data and none of these stood out as a single best candidate to reuse in evaluation testing. Re-deriving these transformations with our evaluation dataset would introduce bias and thus defeat the point of having an evaluation set; instead, for the purpose of our evaluation testing, using the parameters acquired during the development optimization, we recomputed all required transformations across the full 100 samples of our development dataset and used these newly derived transformations in each of our tests as required.

We performed two different experiments to provide a general assessment of the strength of our optimal biometric system configurations. For our first experiment we calculated the EER for each of our best performing normalized and non-normalized classifiers from chapter 6 over our entire evaluation dataset. This meant each of our tests involved

performing classification against both shoe types and in some cases our classifiers were trained with samples from multiple shoe types; as a result of this, we would expect performance to be slightly worse than might be expected were the training and testing performed with a single shoe type (see section 7.3 for more). The results obtained after performing this experiment are demonstrated in table 7.1. These results revealed a clear decrease in performance when compared with the results obtained over the development dataset; this decrease was particularly pronounced over our geometric and spectral feature spaces, suggesting a higher degree of preprocessor overfitting occurred in the optimization of these spaces. With regards to classifier performance, results were more varied; the LDA classifier performed best when compared to all others, while the SVM classifier was a bit weaker than would be expected, perhaps also owing to parameter overfitting on the development dataset. In figure 7.1 the differences in classifier performance can be seen more clearly with the DET curves for the best performing classifier configurations plotted against each other.

Evaluation Dataset Results

	KNN	MLP	SVM	LDA	LSPC
Geometric	4.9711	5.2838	4.463	4.4909	4.2843
Normalized Geometric	5.9668	4.785	4.7105	3.7316	5.399
Holistic	4.7515	3.4412	4.0201	3.19	4.9339
Normalized Holistic	5.6858	4.8669	6.6368	4.1689	4.9078
Spectral	20.2084	16.9458	21.1427	15.416	22.3115
Normalized Spectral	11.6936	8.5892	11.7462	5.9259	12.8159
Wavelet	3.1546	3.1081	2.9145	1.7699	2.4046
Normalized Wavelet	2.8475	2.0137	4.2099	1.6434	2.4325

Table 7.1: This table compares the EER percentages obtained for each of the best classifier-preprocessor combinations from chapter 6, using the cross validated testing scheme described in this section.

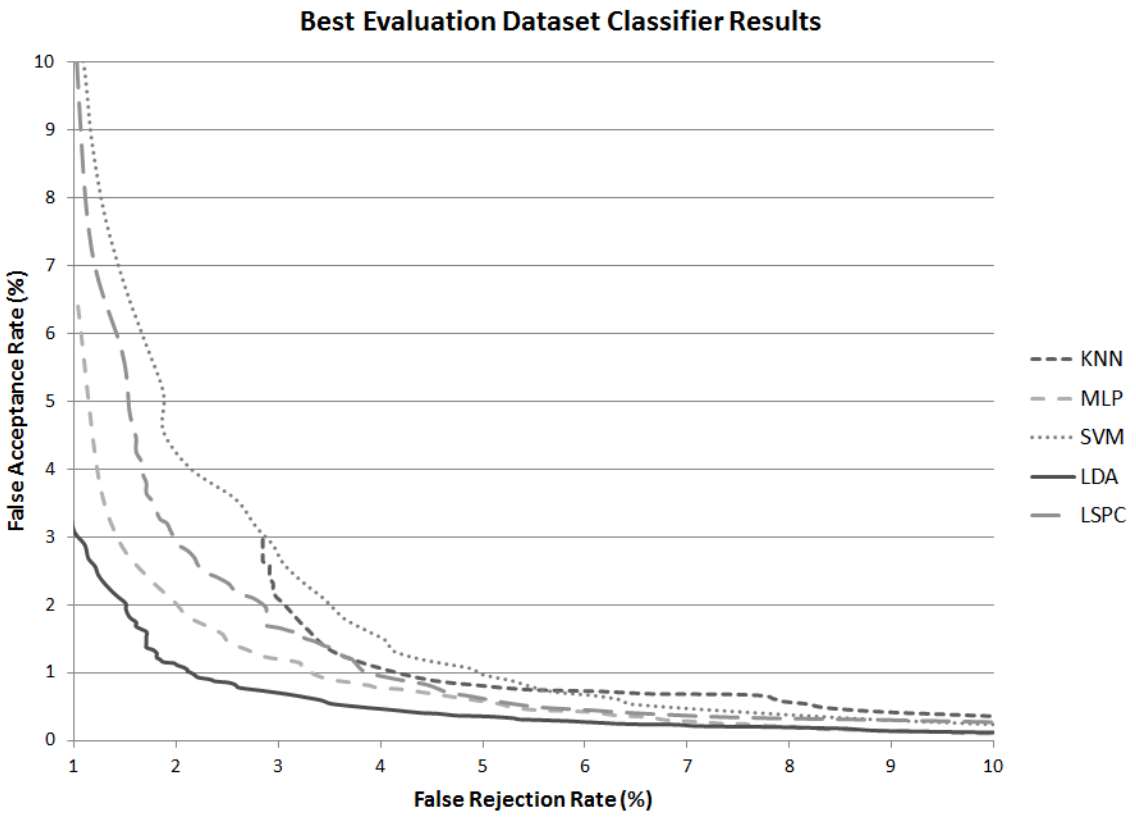

Figure 7.1: This figure demonstrates the DET curves obtained in calculating the EER for the top performing classifier-preprocessor combinations from table 7.1. In this figure the KNN result appears to be cut-off; this was due to a sharp near-zero threshold, which made it difficult to get a smooth error rate curve when approaching a FRR of zero.

In our second experiment we took the best feature space-classifier combinations from table 7.1 and trained them using a varying number of samples per subject. To continue using our chosen cross validation technique this experiment required that at least two steps be used during training, otherwise in some cross validation folds our subject m22_1 would not be assigned any training samples, breaking the test; this training sample requirement also was bounded by our LDA classifier, which required at least two samples be present to correctly derive its inter-subject variance properties. Consequently, we tested the GRF recognition performance with the number of training steps starting at 2

and increasing to 10. When plotted (figure 7.2) the relationship between the number of training steps and the GRF recognition performance became obvious, with the classifiers achieving a sharp increase in performance up to around steps 4 through 6, with slower performance increases thereafter. This, together with the fact that some of our biometric system configurations may have overfit the development dataset, could open the possibility of significantly better GRF recognition results in larger development datasets (i.e. one with enough sample variety to mitigate optimization overfitting) and/or evaluation datasets with a greater number of training samples per subject available.

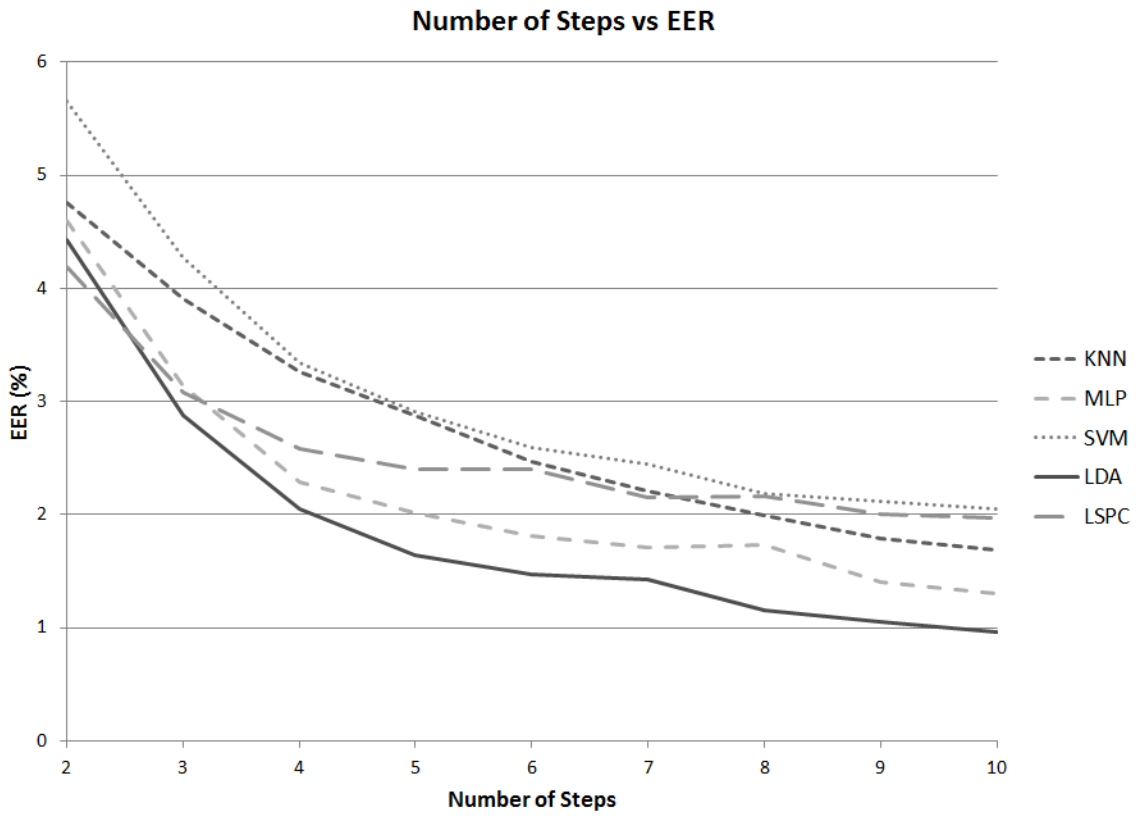

Figure 7.2: This figure demonstrates how the EER for each of our best performing classifier-preprocessor combinations changes with the number of footsteps used to train the classifier.

7.2 Stepping Speed Normalization

In our problem statement we set out to prove the assertion that a relationship useful for footstep GRF-based person recognition exists between stepping speed and the shape of the GRF force signature. In chapter 5 we introduced several normalizers (LTN, LLSR, and LLSRDTW) designed to transform samples with varying step durations to the forms they would be expected to take were they all captured on a common scale with regards to step duration. Two of these normalizers, the novel LLSR and LLSRDTW normalizers, went a step further than simply performing a common uniform transformation and attempted the acquisition of higher resolution temporal-force relationship models parameterized by the total step duration. The preliminary results calculated over our development dataset in chapter 5 indicated that the stepping speed-force signature relationship could indeed be used to increase recognition performance; however, as discovered in the previous section, our development results were subject to an optimization bias and conclusions might not hold for real world data. In this section we put these normalizers to the test against the previously unseen data of our evaluation dataset to better assess our problem statement assertion.

For the purpose of this experiment we took the best biometric system configurations for each step duration normalizer/feature space pair and measured their performance across our entire evaluation dataset, once again implementing the cross-validated testing strategy described in the previous section. To avoid any potential optimization biases we stuck to only computing and comparing results using a KNN classifier with a *K* value of

5, as this was classifier for which all normalizers in chapter 5 were optimized. In this experiment we divided our results according to feature space and presented them in the form of DET curves. This representation of the results allowed us to better identify comparative performance in contrast with what could be identified from the EER alone; namely in some cases the point at which the EER is calculated may be distorted by anomalously high or low value spikes in values that would not be reflective of performance we would normally expect, a classifier with strong performance would be expected to have its DET curve generally fall under that of the others in addition to having a low EER value.

The first set of results we calculated for the purpose of this experiment was a comparison of our step duration normalizers over the geometric feature space. It must be noted that the LLSRDTW normalizer was not applicable to the geometric feature space because the heuristically obtained geometric features required no additional alignment to perform regression comparisons. The DET curves that resulted from applying the remaining normalizer to the geometric feature space are demonstrated in figure 7.3. These results appear to strongly contradict our findings from chapter 5 where the same normalizers were applied to our development dataset and normalization led to a substantial increase in recognition performance over the non-normalized data. This might suggest a higher degree of development dataset overfitting occurred in the normalized spaces over the non-normalized spaces, but it could also suggest that the step duration-based geometric features played a larger role in discriminating subjects in the evaluation dataset.

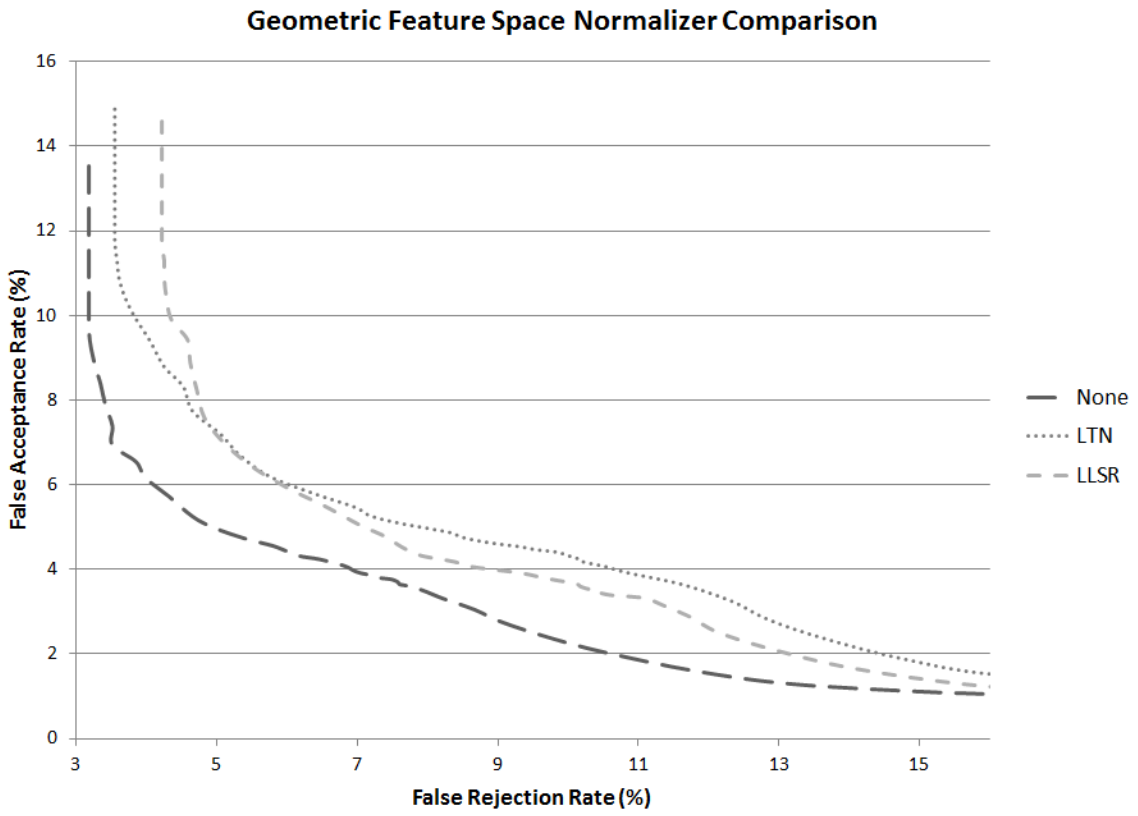

Figure 7.3: This chart compares the DET curves obtained by performing KNN classification on our evaluation dataset over the normalized and non-normalized geometric feature spaces from chapter 5.

Our second set of results was calculated by rerunning the aforementioned normalizer performance comparisons on our holistic feature space, this time with the applicable LLSRDTW normalizer. These results came out more in line with what we expected. The DET curves in figure 7.4 demonstrated a clear increase in GRF recognition performance when any of the step duration-normalized classifiers were applied. The fact that this was in contrast to our geometric feature space results may be due to the fact that the holistic feature set contained no features directly measuring the temporal properties of the data. These results also stood in contrast to the normalized holistic result from the previous section. In our general evaluation dataset analysis, normalization was performed using an

L^1 amplitude normalizer, the best performing holistic feature space normalizer over our development dataset. The results obtained in this section would seem to imply that step duration-based normalizers can achieve better performance than amplitude scaling normalizers when applied in more complicated datasets like our evaluation set.

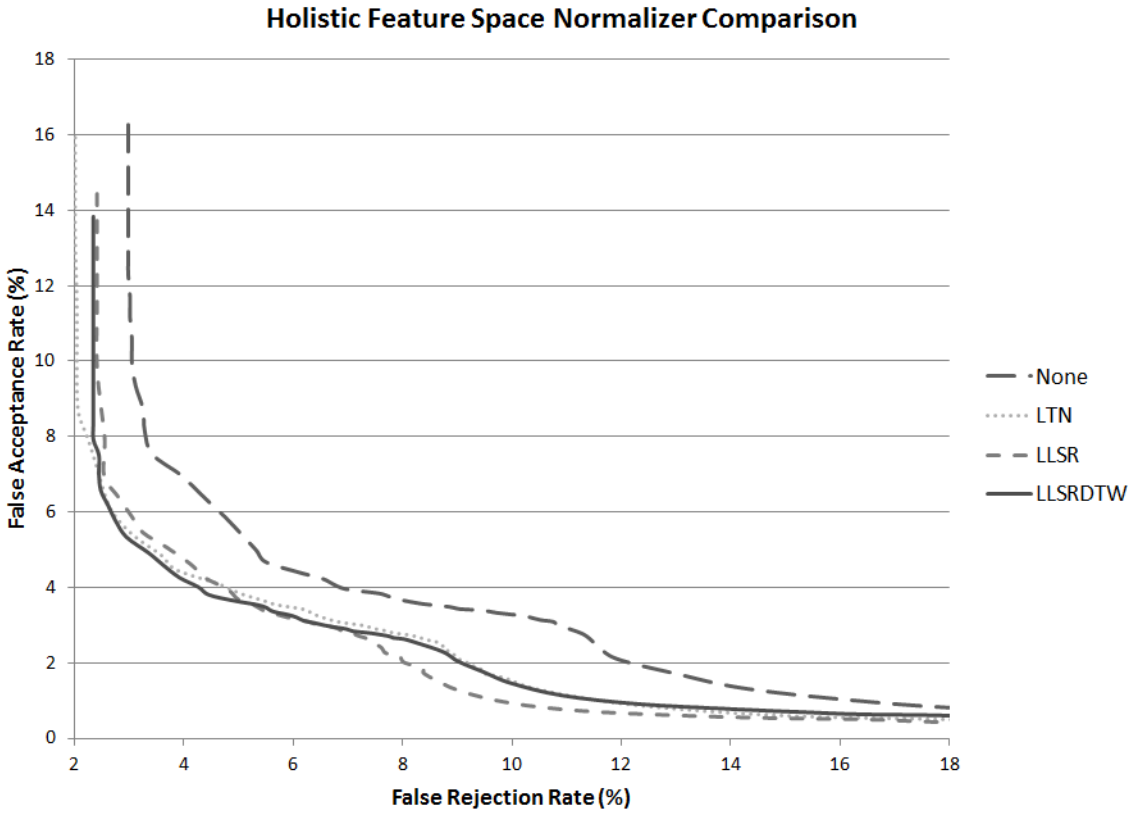

Figure 7.4: This chart compares the DET curves obtained by performing KNN classification on our evaluation dataset over the normalized and non-normalized holistic feature spaces from chapter 5.

The third set of results we collected was taken from our spectral feature space. In this case the LTN normalizer was not applicable as the time dimension upon which such normalization is applied gets negated when the derivative is acquired during the transformation to our spectral feature space. The DET curves that resulted from running

our cross validated test strategy on the remaining normalizers are demonstrated in figure 7.5. As was the case for our general experiments in the previous section, the results obtained from our spectral feature space in this section proved to be far worse than the equivalent results on our development dataset. However, although none of our results were particularly strong, the results obtained when a step duration-based normalization technique was applied again proved to be substantially better than those when no normalization was applied.

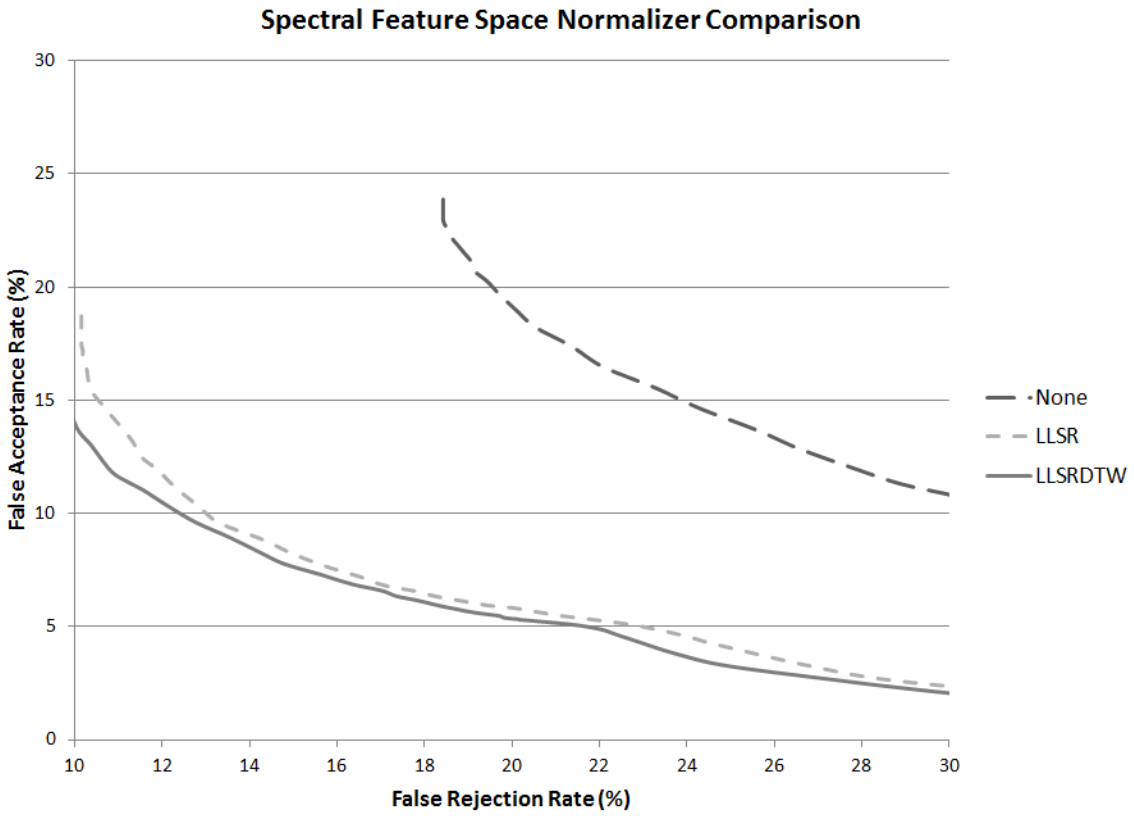

Figure 7.5: This chart compares the DET curves obtained by performing KNN classification on our evaluation dataset over the normalized and non-normalized spectral feature spaces from chapter 5.

For our final performance comparison we compared the performance of our normalizers when applied to our wavelet feature space. The results obtained, shown in figure 7.6, were considerably stronger than those obtained in other feature sets. Moreover, we once again found all of our step duration-based normalizers led to a significant increase in footstep GRF recognition performance when compared with the performance obtained when no normalizer was used.

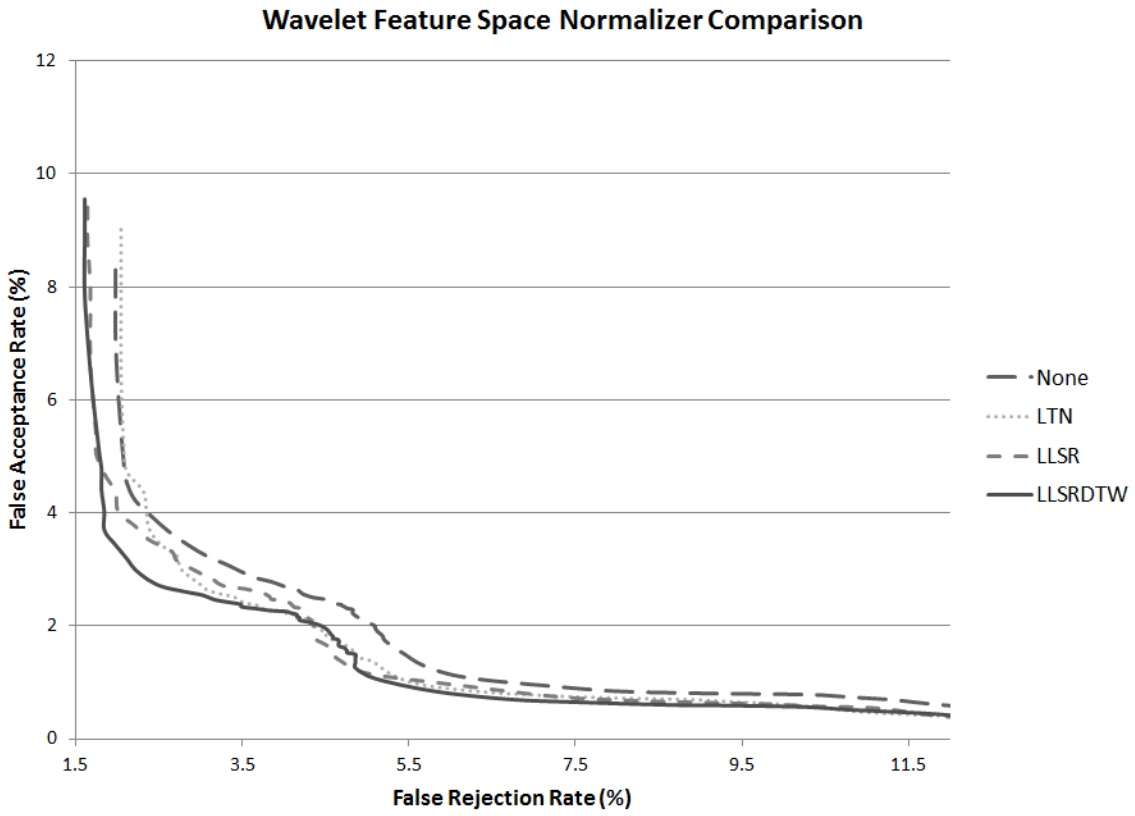

Figure 7.6: This chart compares the DET curves obtained by performing KNN classification on our evaluation dataset over the normalized and non-normalized wavelet feature spaces from chapter 5.

In table 7.2 we did a side-by-side comparison of the EERs acquired from each of the examined feature space-normalizer combinations. Interestingly, while normalization was

of no help when applied to our heuristically derived geometric features, all of our step duration-based normalization approaches led to an improvement in performance when applied to our machine learning derived feature sets. These findings back for our assertion that normalizing for stepping speed can help improve GRF recognition performance, at least when done on a non-heuristically derived feature space, opening up the potential for further gains for normalization models that better fit the relationship between force signature and step duration.

Stepping Speed Normalization Results

	None	LTN	LLSR	LLSRDTW
Geometric	4.9711	5.9743	5.9668	-
Holistic	4.7515	4.3606	4.3197	4.0889
Spectral	20.2084	-	11.9114	11.6936
Wavelet	3.1546	2.8699	2.9536	2.6633

Table 7.2: This table compares the ERR percentages obtained for each of the feature space-normalizer combinations tested in this section.

7.3 Shoe Type Variation

When assessing the performance that might be expected of GRF-based footstep recognition in a real world scenario it would typically be unreasonable to assume that the people using the system never change their footwear. Instead we might expect footwear to change based on weather, formality of the occasion, or simply due to a normal shoe-replacement cycle. In creating a biometric system for the purpose of footstep recognition it is important that we understand the implications behind working in an environment that may encounter people who enroll with the system using one shoe and later attempt to verify using a different shoe. In our problem statement we made the assertion that variation in shoe type will have a negative impact on recognition performance. In this section we test that assertion and aim to gain a better understanding of the biometric system configurations that best account for shoe variations. To accomplish this we have regenerated the results matrix from section 7.1 (table 7.1), but this time using four different subsets of the evaluation dataset.

Our evaluation dataset was divided by shoe type, about half the samples were collected with subjects wearing a Verona runner and the other with subjects wearing an Orin runner. To provide a full assessment of the impact of shoe type across the dataset we devised the experimental test strategy illustrated in figure 7.7. Under this strategy we used the transformations previously acquired in our development dataset (with subjects wearing Asics runners) to perform all data preprocessing required for our evaluation testing. We already obtained the results combining both shoe types for training/testing

(Combined Results) in section 7.1, so for the remainder of this section we demonstrate the additional results obtained using subsets of the data across only a single shoe type (Verona Results and Orin Results) and those where the tested shoe type always differed from the trained shoe type (Verona-Orin Results and Orin-Verona Results).

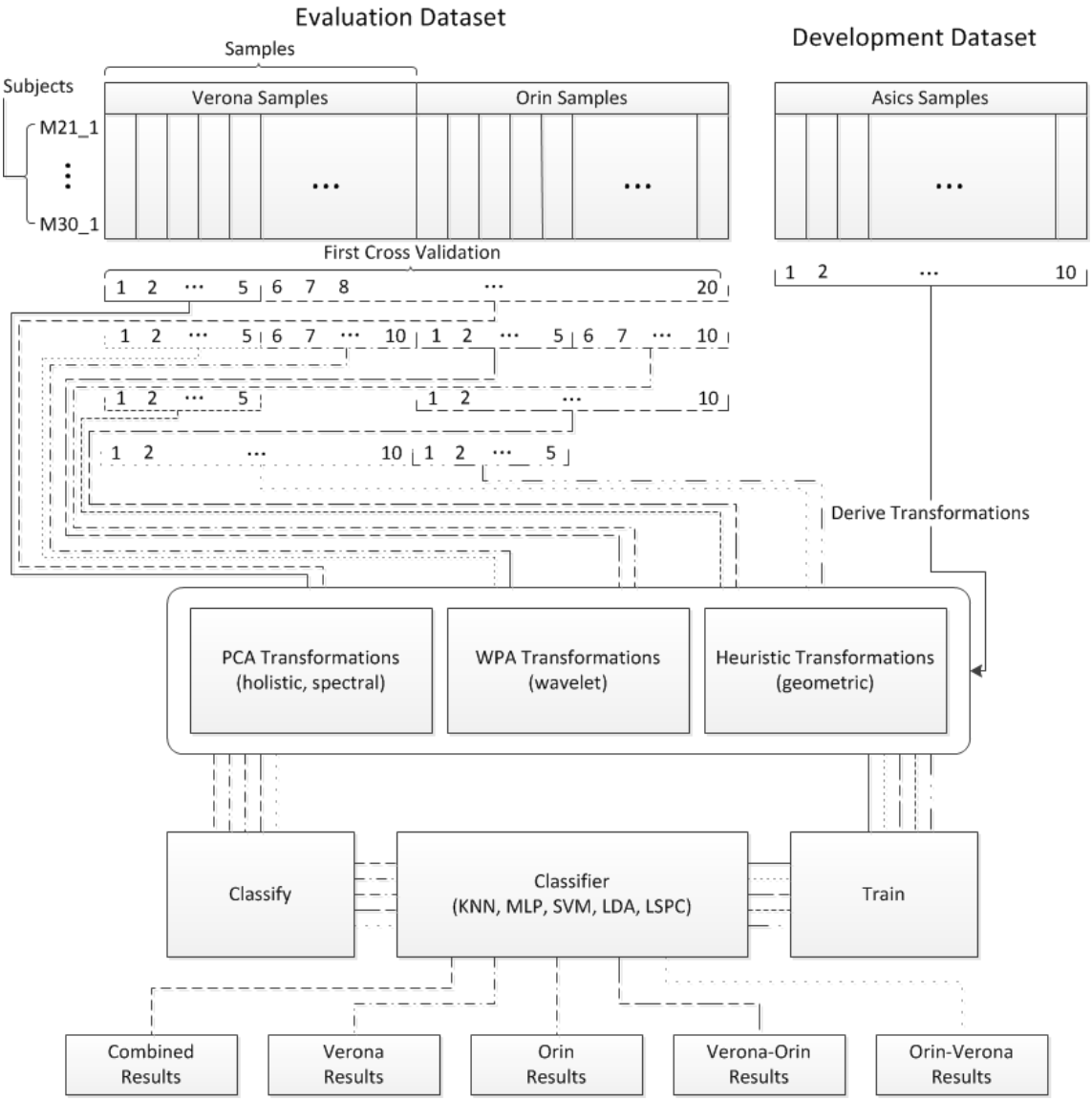

Figure 7.7: This figure demonstrates the test strategy used to obtain results for our various shoe-based result sets. Here we provide an example of a single cross validation with the training/testing samples identified according to their alignment to the evaluation samples.

The first results we collected involved classifiers trained with footsteps from the same shoe type as the type on which they were tested. In this experiment we reapplied the cross validated classifier-preprocessor calculations from section 7.1 to the 100 Verona samples (table 7.3) and 99 Orin samples (table 7.4) of our evaluation dataset; again, in the case of the missing Orin sample several cross validations included a subject trained with only 4 rather than 5 samples. Due to the inclusion of cross validations with a missing Orin sample we could expect a slight negative bias in the results obtained from that shoe type. Our expectation was also that overfitting of preprocessors and classifier parameters would result in weaker recognition performance in these sample sets when compared with our development dataset results; however, as can be seen in our Verona results table, in the case of the Verona shoe type the normalized wavelet results actually appeared to be nearly as strong and, in some classifiers, stronger than our development results.

Verona Dataset Results

	KNN	MLP	SVM	LDA	LSPC
Geometric	3.4222	5.1	3.7333	4.8555	4
Normalized Geometric	3.8666	4.7555	4.1222	6.7333	3.0222
Holistic	3.8333	2.3777	3.1	2.8111	3.7333
Normalized Holistic	4.8666	2.8555	5.7	2.9666	4.2
Spectral	10.8222	10.1666	10.0888	8.6444	11.4555
Normalized Spectral	6.1888	3.0333	3.4	1.6777	5.7555
Wavelet	1.8666	1.4777	1.2	0.8111	1.0666
Normalized Wavelet	1.4333	0.8	2.4555	0.2666	0.9777

Table 7.3: This table compares the EER percentages obtained for each of the best classifier-preprocessor combinations from chapter 6 when applied to the Verona shoe type samples in our evaluation dataset.

Orin Dataset Results

	KNN	MLP	SVM	LDA	LSPC
Geometric	5.0505	7.0145	4.4893	5.1402	4.2873
Normalized Geometric	6.3636	6.5881	5.7126	4.3995	6.4309
Holistic	4.4893	3.1537	3.3557	2.3681	4.2985
Normalized Holistic	4.2312	2.9629	5.1627	3.4118	3.1088
Spectral	16.2177	12.9854	18.6644	11.358	18.1144
Normalized Spectral	10	7.7216	10.8193	4.9943	13.0639
Wavelet	3.1537	3.7485	3.0751	2.1773	2.2109
Normalized Wavelet	1.8406	2.1548	3.5802	2.0875	1.1616

Table 7.4: This table compares the EER percentages obtained for each of the best classifier-preprocessor combinations from chapter 6 when applied to the Orin shoe type samples in our evaluation dataset.

Having collected our results across single-shoe subsets, our next experimental results were obtained by training our classifiers with the samples from one shoe and testing them with the samples from the other. In this case the Verona-Orin result set (table 7.5) refers to results obtained by performing 10-fold cross validated training with 5 samples per fold in the Verona sample set, while running each trained fold against the entire Orin sample set to calculate our EER results. The Orin-Verona result set (table 7.6), on the other hand, refers to the opposite, with Orin samples used for training and Verona samples used for testing. Again the Orin sample set may have suffered a slight negative bias due to a missing training sample in several cross validations; however, in our findings the Orin-trained results often proved better than the Verona-trained results, in comparison with the previous single-shoe results where the Verona-trained results were almost universally better. Our primary expectation was that, on a whole, the results obtained by these mixed-shoe evaluation subsets would be worse than those obtained via the single-shoe evaluation subsets and this seems to have held true in these results.

Verona-Orin Dataset Results					
	KNN	MLP	SVM	LDA	LSPC
Geometric	4.9214	6.1054	5.6509	5.3928	4.6913
Normalized Geometric	4.5111	4.4781	3.872	4.5735	4.8484
Holistic	4.624	3.6251	4.0235	3.973	3.9057
Normalized Holistic	5.4826	5.3086	6.2457	5.0392	4.9775
Spectral	23.3277	21.8013	23.5465	20.5723	25.0841
Normalized Spectral	14.1526	13.3838	15.3367	11.5375	16.8855
Wavelet	3.7149	4.45	4.0123	2.6206	3.0303
Normalized Wavelet	3.4511	3.0303	7.0314	2.2334	3.0808

Table 7.5: This table compares the EER percentages obtained for each of the best classifier-preprocessor combinations from chapter 6 when trained with our evaluation Verona shoe samples and tested against our evaluation Orin shoe samples.

Orin-Verona Dataset Results					
	KNN	MLP	SVM	LDA	LSPC
Geometric	6.1833	5.2722	4.5444	4.0833	4.6722
Normalized Geometric	5.9888	4.1611	5.0333	3.0444	5.1777
Holistic	4.5944	3.2333	4.1444	2.6666	5.7833
Normalized Holistic	6.1277	5.8722	7.4	4.4388	5.4055
Spectral	26.6611	23.1166	28.9166	19.0444	27.75
Normalized Spectral	14.5388	10.1166	15.2777	6.7166	13.3388
Wavelet	4.6722	3.6833	4.2388	1.6833	3.1944
Normalized Wavelet	3.7888	2.4722	4.7166	1.5888	3.1

Table 7.6: This table compares the EER percentages obtained for each of the best classifier-preprocessor combinations from chapter 6 when trained with our evaluation Orin shoe samples and tested against our evaluation Verona shoe samples.

The EER results from the tables presented in this section would appear to back our assertion that shoe type variation can have a negative impact on footstep GRF-based person recognition performance. To further demonstrate this, in figure 7.8 we plotted the DET curves for our normalized-wavelet recognition performance over the combined (section 7.1), single shoe, and cross shoe subsets. In this figure we have plotted the

results for each classifier type, with each appearing as a separate line identifying the shoe subset from which it was obtained. While there tended to be some overlap between these evaluation subsets results due to classifier differences, on a whole we see the solid line-single shoe results tended towards lower EER values, the dashed line-cross shoe results tended toward higher EER values, and the double line-combined results tended to be somewhere in between. The position of the combined shoe type result curves would also support a finding in [5], which suggested training with multiple shoes per subject can improve recognition performance.

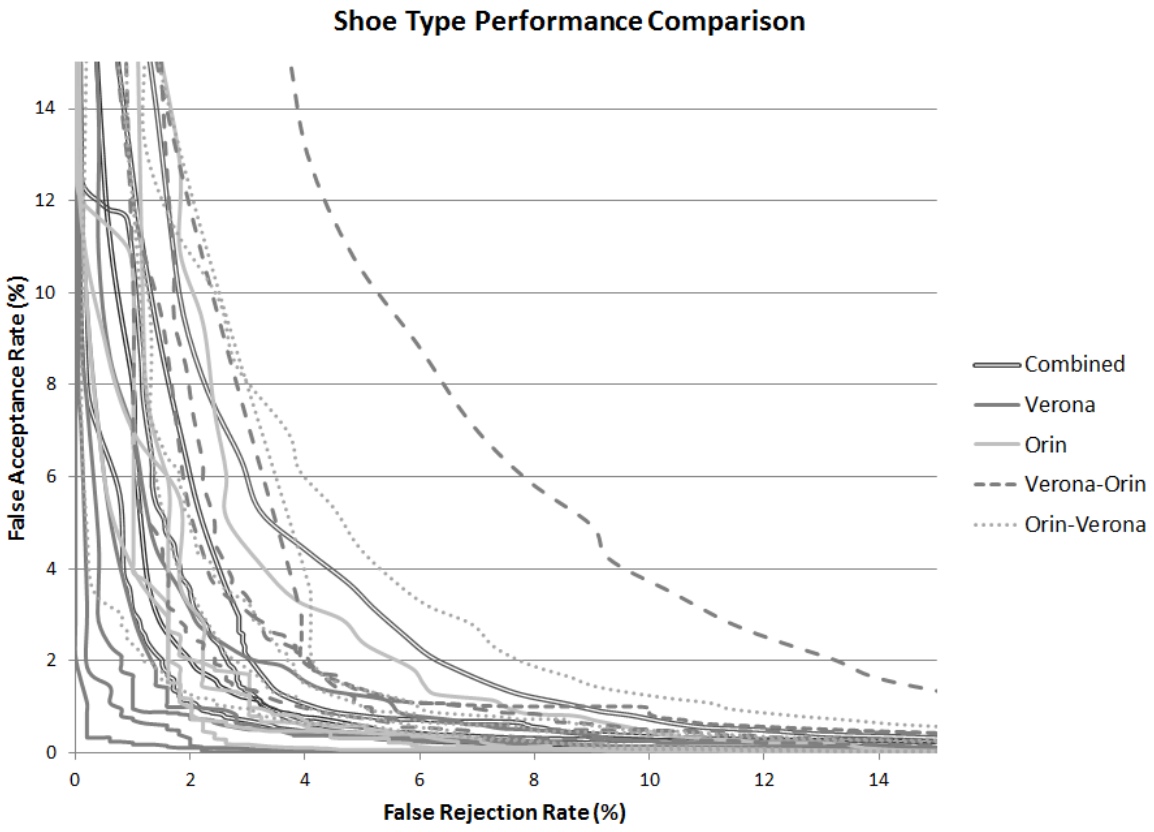

Figure 7.8: This figure compares the DET curves for each classifier-shoe subset pair for our normalized wavelet preprocessor. In this case each line represents a different set of classifier results with the line styling indicating shoe subset from which it came.

7.4 Summary

This chapter examined the results obtained after taking some of the top performing biometric recognition system configurations from chapters 4 through 6 and applying them to our evaluation dataset. While many of our results proved weaker than those obtained from our development dataset (the set on which our preprocessors and classifiers were optimized) we still found that reasonably high GRF recognition rates could be obtained. This was particularly true for our wavelet feature space, which outperformed all other feature spaces. We also found that our geometric and spectral feature spaces generated results that were substantially weaker than those obtained from our development dataset, suggesting those two techniques were not very good at generalizing the recognition problem for more complicated previously unseen datasets.

With regards to our two problem statement assertions, preliminary analysis of our results appeared to back both. We re-applied the step duration-based normalizers (LTN, LLSR and LLSRDTW) from chapter 5 to our evaluation data and found that, in all but our geometric feature space, each of these led to improved recognition results. Moreover, when comparing results obtained from testing against a classifier trained on samples from a common shoe type and those obtained from the same classifier tested against samples from a different shoe type, we found that the former generally performed better than the latter. In the next chapter we further explore the backing for these assertions, analyze how our results compare with those of related research, and discuss areas that may lead to future improvements in the field.

Chapter 8

Experimental Analysis

The problem statement in chapter 1 outlined two objectives for which we devised the experiment presented in the preceding chapters. Our primary objective was to address current research gaps regarding the effect of normalization and shoe type variation when performing footstep recognition via GRF signatures; we went on to refine this objective to refer to the goal of verifying the two assertions: that shoe variation can have a negative impact on GRF recognition performance and that stepping speed-based normalization could be used to improve GRF recognition performance. Our secondary objective was to expand upon the work done in previous GRF-recognition studies with respect to feature extraction and classification; in this objective we aimed to compare various previously untested footstep GRF-based biometric systems and obtain results that could be used to make comparisons with the results obtained in related work. It was hoped that this work would contribute to the present day understanding of GRF recognition and address some of the technical issues facing the deployment of such a system in a real world setting.

In this chapter we begin by providing a breakdown of our findings as they related to our problem statement objectives. We continue on to discuss the implications behind these findings. Finally, we explore ways in which we may be able to improve upon the results discovered.

8.1 Findings

The results collected in chapter 7 appeared to support our problem statement assertions. A closer inspection also reveals that many of our results were consistent with findings in related work, with a few exceptions. To meet our outlined objectives, in this section we further analyze these results and compare them with both our own experimental expectations and the findings of related research. We begin by presenting a more detailed look at how closely our results adhered to our problem statement assertions, and then identify the areas in which our recognition results supported or contrasted the results of related research, with special attention given to our never-before-evaluated GRF-based biometric system configurations.

8.1.1 Shoe Type

To help verify the assertion that variation in shoe type can have a negative impact on GRF recognition performance the previous chapter presented results for biometric system configurations trained and tested with the same shoe type, trained and tested with varying combinations of shoe types, and trained with one shoe type then tested against a different shoe type. Further analyzing these results the relationship between shoe type variation and GRF-recognition performance becomes clear. In table 8.1 we compare the averaged and best classifier performance achieved across our evaluation dataset for each of our shoe type training configurations. Looking at the demonstrated comparisons, on a whole, the same-shoe training/testing results were stronger than the other two, while the results that included combined shoe types during training generally outperformed those that

were obtained by always testing against a different shoe type from the one upon which the training was performed. A notable exception to this tendency was observed in the results over our geometric feature space, which demonstrated less performance variance across shoe types and was even shown to perform better in some cases when trained and tested across differing shoe types. This result was interesting because it is consistent with the findings of [31], in which analysis was performed over a geometric feature space and it was concluded that differences in footwear did not significantly impact GRF recognition performance.

Shoe Variation Findings

	Same Shoe Testing		Combined Shoe		Cross Shoe Testing	
	Averaged	Best	Averaged	Best	Average	Best
Geometric	4.7092	3.8547	4.6986 **[-0.2]**	4.2843 **[+11.1]**	5.1517 **[+9.3]**	4.3873 **[+13.8]**
Normalized Geometric	5.1994	3.7108	4.9185 **[-5.4]**	3.7316 **[-0.5]**	4.5688 **[-12.1]**	3.4582 **[-6.8]**
Holistic	3.352	2.3729	4.0673 **[+21.3]**	3.19 **[+34.4]**	4.0573 **[+21]**	3.1458 **[+32.9]**
Normalized Holistic	3.9466	2.9092	5.2532 **[+33.1]**	4.1689 **[+43.3]**	5.6297 **[+42.6]**	4.7081 **[+61.8]**
Spectral	12.8517	10.0012	19.2048 **[+49.4]**	15.416 **[+54.1]**	23.982 **[+86.6]**	19.8083 **[+98]**
Normalized Spectral	6.6654	3.336	10.1541 **[+52.3]**	5.9259 **[+77.6]**	13.1284 **[+84.9]**	9.127 **[+173.5]**
Wavelet	2.0787	1.4942	2.6703 **[+28.4]**	1.7699 **[+18.4]**	3.53 **[+68.8]**	2.1519 **[+44]**
Normalized Wavelet	1.6757	0.7141	2.6294 **[+56.9]**	1.6434 **[+130.1]**	3.4493 **[+105.8]**	1.9111 **[+167.6]**

Table 8.1: This table presents the EER percentages obtained by comparing shoe type variations in chapter 7 (with percent differences from the "same shoe" row results shown in the bold square brackets). The averaged results were collected across our 5 different classifiers and the "best" classifier result for "same shoe" and "combined" shoe refers to the average of the two best individual classifier results from the Orin and Verona shoe types.

In the related GRF recognition studies we found three other studies describing the use of multiple shoe types in their dataset. These studies included [3, 4], which used multiple shoe types per person in their dataset but did not examine the influence of such variations on their results, and [5], where it was suggested that variation in shoe type between training and testing would have a negative impact on GRF performance. In [5] no direct measurement of GRF-recognition performance was used to assess the impact of shoe type, however, the conclusion that was reached would also appear to be consistent with our results; in that case results were acquired using a holistic feature extraction technique, a technique for which we observed a substantial decline in recognition performance when training/testing shoe variation was introduced. On average our recognition results across the full set of preprocessors and classifiers came out about 50% worse when tested across shoe types, and, while it may be possible to generate feature spaces less susceptible to shoe variation, like our normalized geometric feature space turned out to be, these spaces ended up producing results that were far worse than our better performing machine learning-based extraction techniques. In view of this we believe that our findings, in a general sense, do in fact verify the assertion that shoe type variation between training and testing will negatively impact GRF recognition performance, but with a noteworthy caveat that will be explored in subsection 8.2.1.

8.1.2 Normalization

In addition to providing a backing for our shoe variation assertion, chapter 7 also presented a set of results to help verify our second problem statement assertion regarding

stepping speed. In that assertion we predicted a relationship between stepping speed and the GRF force signature could be obtained and successfully applied to improve GRF recognition performance. This assertion had support from the work of a MV-based gait study [14], in which the use of the LTN normalizer led to improvements in recognition performance from 8-20%. In our research we expected a similar normalization technique, this time based on step duration, could be used to increase performance. We tested four different normalizers including two novel ones developed for the purpose of our research (LLSR and LLSRDTW). The results obtained are shown in table 8.2.

Development Normalization Findings

	None	LTN	LLSR	LLSRDTW
Geometric	1.3333	1.0333 **[-22.5]**	0.1777 **[-86.6]**	-
Holistic	2.5555	2.3 **[-9.9]**	2.5333 **[-0.8]**	2.3 **[-9.9]**
Spectral	2.0222	-	2.6 **[+28.5]**	1.8444 **[-8.7]**
Wavelet	1.2888	1.1 **[-14.6]**	2.3333 **[+81.1]**	1.4555 **[+12.9]**

Evaluation Normalization Findings

	None	LTN	LLSR	LLSRDTW
Geometric	4.9711	5.9743 **[+20.1]**	5.9668 **[+20]**	-
Holistic	4.7515	4.3606 **[-8.2]**	4.3197 **[-9]**	4.0889 **[-13.9]**
Spectral	20.2084	-	11.9114 **[-41]**	11.6936 **[-42.1]**
Wavelet	3.1546	2.8699 **[-9]**	2.9536 **[-6.3]**	2.6633 **[-15.5]**

Table 8.2: These tables demonstrate a comparison of the EER percentages obtained after applying various step-based normalizers to our four feature spaces (with the percent differences from the non-normalized row results shown in the bold square brackets).

The improvement in recognition performance observed for many of our step-based normalizers was in line with the improvement in performance observed for the MV-based gait recognition in [14]. Moreover, while no other previous GRF-based recognition studies examined the effect of such normalizers directly, in [7] the dataset was broken up according to three categories of stepping speed (low, normal, and high) and experiments were performed to compare the effect of stepping speed on GRF recognition performance. What they found was a decrease in performance of about 9-15% when testing a wavelet feature space-SVM classifier biometric system configuration against different stepping speeds from those on which it was trained, and a decrease of about 39-57% when running the same tests using a geometric feature space. Smaller decreases in performance were observed when training samples of mixed speeds were used during training, but the results in [7] did appear to support the idea that there was room for improvement in GRF recognition if the relationship between stepping speed and the GRF signature could be modelled with some degree of accuracy.

Although the results demonstrated in [7] were acquired using a LTN normalizer, no direct comparison was made over non-normalized data so it is not known whether their application of normalization led to any improvements. In our own results we did observe an evaluation performance improvement across most of our feature spaces when LTN and other step duration-based normalizers were applied using our KNN classifier; this, however, was not the case for our geometric feature space, in which a decrease in performance was observed, likely owing to development dataspace overfitting. It must also be noted that, of all the normalization approaches examined, our new LLSRDTW

normalizer was found to deliver the best overall evaluation performance increase, leading to an increase in GRF recognition performance of about 14-15% in our better performing holistic and wavelet feature spaces. Unlike the LTN normalizer, which performed a simple linear scaling operation on the data, the LLSRDTW normalizer was designed to dynamically model the relationship between GRF force signature and stepping speed. Therefore, because it led to improved recognition performance over the LTN normalizer, we have come to the conclusion that a modeled relationship between step duration and the GRF signature can in fact be utilized to achieve better recognition results, satisfying the second of our problem statement assertions.

8.1.3 Biometric System

As a secondary objective in our problem statement we aimed to expand upon work done in previous GRF recognition studies with respect to feature extraction and classification. Back in section 3.1 of chapter 3 we presented a breakdown of our studied feature extraction, normalization, and classification techniques according to their use in previous GRF recognition studies. While many studies examined one or more of our chosen biometric system configurations, none of the previous research compared recognition performance over the entire set. In the previous subsection we demonstrated our findings with regards to the normalization component of our biometric system. While normalization was an area of focus in this thesis, we also uncovered some interesting findings when analyzing our various feature extraction and classification techniques.

Feature Space Results

	Averaged Development Classifer Results	Averaged Evaluation Classifier Results
Geometric	2.0911	4.6986 **[+124.6]**
Normalized Geometric	0.9377	4.9185 **[+424.5]**
Holistic	1.6355	4.0673 **[+148.6]**
Normalized Holistic	1.6266	5.2532 **[+222.9]**
Spectral	1.5244	19.2048 **[+1159.8]**
Normalized Spectral	1.2577	10.1541 **[+707.3]**
Wavelet	1.4221	2.6703 **[+87.7]**
Normalized Wavelet	1.131	2.6294 **[+132.4]**

Table 8.3: This table compares the features space-influenced EER percentage results averaged across our five different classifiers. The values in the bold square brackets represent the same-row percent difference between the two datasets.

In table 8.3 a comparison of our various examined feature extraction techniques is presented for results that have been averaged across all tested classifiers. One of the most obvious disparities in the above results was the substantial decline in recognition rates when comparing the development and evaluation spectral feature space results. Interesting, this decrease in performance appears to be strongly influenced by the evaluation use of different training and testing shoes, because the chapter 7 results for single shoe type training and testing subsets proved considerably better. It stands to reason that some characteristics specific to shoe type may have negatively influenced the selection of principal components during the derivation of the spectral transformation on our development dataset. Aside from the spectral results, we observed a general decrease in recognition performance between the development and evaluation dataset of approximately 100-200%. The decrease in recognition performance between the evaluation and development datasets was far greater than those observed in [3, 4], which

also broke data into separate development and evaluation datasets, however our datasets contained much more information per sample and overall our results were far stronger than the ones acquired in those studies.

Due to the differences in GRF sample characteristics between our data and that of previous studies, we were not able to directly compare our results with related research. Instead, to gauge our feature space performance in the context of related work, we decided to assess our results via examining the relative performance differences found when results were acquired for two or more feature spaces in the related studies. With regards to the holistic feature space, in our experiment we observed a 10% improvement in GRF recognition when we compared our holistic feature space with our geometric space (using a SVM classifier). The same configuration led to a 21% improvement in recognition results when examined in [4]. In [32] more combinations of feature space-classifier pairs were compared with demonstrated improvements of 35%, 31%, and 27% when geometric feature spaces were used over spectral feature spaces for the KNN, MLP, and SVM classifiers, respectively; in our evaluation results we discovered improvements of 75%, 68%, and 78% for the same respective feature space-classifier pairs. Moreover, in [7] the use of a wavelet feature space led to an improvement in recognition results of over 25% when compared against a geometric feature space (via a SVM classifier); this result turned out to be similar to our own, in which case the same configuration led to a 34% improvement in recognition performance. So, while the results in [4] and [32] might suggest the holistic or geometric feature spaces were best suited for GRF

recognition performance, our comparative performance over the wavelet feature space was noticeably stronger than either.

Classifier Results (ignoring spectral feature space)

	Averaged Development Feature Space Results	Averaged Evaluation Feature Space Results
KNN	1.3462	4.5628 **[+238.9]**
MLP	1.7499	3.9164 **[+123.8]**
SVM	1.4407	4.4924 **[+211.8]**
LDA	1.5277	3.1657 **[+107.2]**
LSPC	1.3055	4.06 **[+210.9]**

Table 8.4: This table compares the classifier-influenced EER percentage results averaged across our geometric, holistic, and wavelet feature spaces. The values in the bold square brackets represent the same-row percent difference between the two datasets.

In contrast with our feature extraction findings, the results acquired from our classifiers demonstrated less variability between one another. These results, shown in table 8.4, compare our two datasets averaged across the geometric, holistic, and wavelet feature spaces; we deliberately left the spectral feature space out from this analysis as it was found to produce outlier results. One interesting finding here was the significant difference in classifier rankings between the evaluation and development dataset. For instance, the KNN classifier, which was second best over the development data, turned out to be the worst performing classifier over the evaluation data. Likewise, the LDA classifier, which was second worst in the development data, turned out to be the best by a wide margin of about 19% in the evaluation data. When compared with the findings in previous multi-classifier GRF recognition studies [3, 32, 7] our averaged evaluation

KNN results were consistent, with KNN performing worse than SVM, MLP, or LDA in those studies. However, our findings differed from [32] and [7] with respect to SVM classifier performance. In [32] the SVM classifier performed about the same as the MLP classifier, while in our findings the SVM was noticeably worse overall (though substantially better over the geometric space). Moreover, in [7], the SVM classifier produced similar performance when compared with the LDA classifier, whereas in our case the SVM classifier performed far worse than the LDA classifier (about 61% worse in the equivalent wavelet feature space). For the purpose of our research we also examined a classifier not used in any of the previous GRF recognition studies, the LSPC classifier. The evaluation performance acquired from this classifier ended up being around the average of the other four classifiers.

In addition to the classifiers themselves, we also examined the performance effects resulting from changes in the number of training samples. By doubling the number of samples in training from 3 to 6 we saw a 37% increase in wavelet space recognition performance, while doubling the number of samples in training from 5 to 10 led to a 32% increase in performance. The improvements in performance from adding training samples became less significant as more samples were added, a finding that was consistent with those of [3], in which performance was found to increase with samples added up to about 40 samples and leveled off thereafter; this would also imply that our results could be improved further by increasing the number of training samples.

8.2 Considerations and Implications

The findings presented in the previous section gave us some insight into the behaviour we could expect from our biometric system in an experimental environment. However, if we were to take our findings and apply them to a real world environment there are a number of caveats that must be considered. Moreover, the characteristics of the results backing our findings and the methods by which they were acquired have important implications for how we would go about selecting an appropriate biometric system configuration for deployment. In the subsections that follow we examine these considerations and implications as they relate to three different aspects of our biometric system.

8.2.1 Data

The practicality of our findings with regards to a real world system is subject to constraints imposed upon it by our chosen dataset. Our dataset differed from any dataset used in previous GRF recognition studies, and thus our results were not directly comparable with those of related research. Several of the other GRF studies with larger datasets used data collected in a more realistic environment with both more subjects and subjects that may not have been as cooperative as our examined subjects. Under that assumption we expect our biometric system performance would be comparatively weaker than the findings might suggest when matched against the results of a study like [4]. Furthermore, our data and analysis had several constraints that may not be realistic in a practical setting; namely, we only examined GRF samples that included a full heel-to-toe step, we only examined walking samples, we did not include imposter attempts from

people not seen during training, and, in our multi-shoe analysis we only examined variations of running shoes (ignoring other potential footwear like boots or sandals). Moreover, in this thesis we did not consider the actual foot (left or right) used in obtaining the samples and further investigation would be needed to assess the effectiveness of our techniques when subject to varying feet used in training and testing. All of these concerns could potentially result in reduced performance in a practical setting; however, when accounting for different aspects of our data, with respect to a real world setting, there are other ways in which our application would have a tendency to produce better results than those discovered in our findings.

One area in which our chosen data may have led to weaker results than those that could be expected in a real world scenario relates to the uniformity in the selected subjects. All of our samples were collected from young athletic males, yet in a practical setting we would expect a larger demographic of subjects with far more inter-subject variability, making it easier, on average, to distinguish different subjects. Additionally, variation in footwear between subjects, as opposed to the within subject variations examined in this thesis, may be more likely in a real world application. If inter-subject variation in footwear turned out to be more common than intra-subject variations then we might expect such variation to improve upon our recognition performance with the choice of footwear helping to distinguish subjects. Finally, there is also reason to believe that the recognition improvement observed in our work with respect to previous GRF studies may have come, at least in part, as a result of the more information rich samples captured; in our work we examined 8 distinct GRF signals with a dimensionality of approximately

12800 points per sample, a larger sample space than any of the previous GRF recognition studies.

8.2.2 Preprocessing

After analyzing our findings with respect to our data preprocessors (normalizers and feature extractors) we were able to identify several factors that may influence results and usability were we to apply our biometric system to a practical setting. One area that we chose not to analyze in this thesis was the relative computational efficiency for each of our preprocessing techniques; yet, if the system were to be deployed in a practical setting we may find that the performance boost gained by performing LLSRDTW normalization could, for instance, be offset by the increase computation time due to its inclusion. Still, there are other ways in which a more practical setting may introduce recognition performance improvements to our findings, particularly with respect to the greater number of samples available. Many of our preprocessors were built upon statistical analysis-based techniques (i.e. PCA, Fuzzy c-means clustering and regression), which we expect would produce more accurate models were more data available for their computation. In that respect, our results may also have potentially been weaker than might be expected in relation to those GRF recognition studies with more available training samples like [3, 4] or [7].

Looking further into our preprocessor findings we were able to arrive at several additional considerations and conclusions. First and foremost we proved that normalizing

for stepping speed could in fact be done to boost recognition performance and we suspect an even greater improvement could be accomplished with the development of stronger stepping speed-GRF signature models. Looking back at our findings, we can also see that the choice of feature extraction technique had more of an impact on performance than the choice of classifier or normalizer, suggesting further research into feature extraction techniques might lead to greater improvements in GRF recognition than research focused on either of the other biometric system components. It should also be noted that many of our preprocessing techniques varied slightly when compared directly with the same techniques in related studies, making direct comparisons with previous research difficult; for instance, in our WP feature extractor we performed wavelet decomposition using the Legendre 04 wavelet function, which we found produced better performance than the Coiflet 06 wavelet function used in [7]. One final interesting implication related to the domain upon which feature extraction was performed. When acquiring our spectral feature space, extraction was done completely within the frequency domain and we found that results suffered when analysis was extended to include multiple shoe types. By comparison the holistic and wavelet feature spaces, computed over a time and time-frequency domain, respectively, resulted in relatively small decreases in performance when extended to multiple shoe types, suggesting a possible weakness in the use of frequency domain analysis when subject to environment variations during GRF acquisition.

8.2.3 Classification

The factors affecting our classification findings were similar to those affecting our preprocessor findings. In a practical setting computational efficiency would likely become more important and classifiers such as the LDA classifier, which involved large matrix transformations, might not overcome the efficiency-performance trade-off. Yet, once again, when analyzed over a larger more realistic dataset we may be able to achieve improved recognition performance; in this case, by performing classifier parameter optimizations over a larger dataset we expect we would be able to derive parameters that better generalize subject classification boundaries. Taking this into account, it again seems that our results could have been weaker with respect to those of previous studies than might be expected in a practical setting. Moreover, some of our classifiers refer to algorithm variants that differ from those studied in related work (i.e. our use of weighted KNN vs. non-weighted KNN in other studies); however, in this case it is hard to tell whether or not these variations improved or decreased our results relative those of the previous GRF recognition studies.

In addition to analyzing classifier performance, this thesis categorized classifiers as being either generative or discriminative, and either eager learning-based or instance-based; we examined classifiers belonging to each of these categories. Further analysis reveals that the category a classifier belongs to may have important implications with respect to how it might perform in a practical setting. Instance-based classifiers, like the KNN classifier, retain all training data in memory and often need to access the entire set of training samples every time a classification is required; consequently, such classifiers may suffer

in terms of both efficiency and memory usage as datasets grow. The use of a generative classifier, like the LDA classifier, may also adversely affect performance. In this category of classifier there is an expectation that some knowledge of class (subject) distribution in the dataset is known a priori, while in a practical setting it may be unfeasible to get a good estimate of the class distribution. Consequently, looking at the classification techniques strictly at a high level, there is reason to believe that eager learning-based discriminative classifiers may be best suited for application to the GRF recognition problem in a practical setting.

8.3 Potential Improvements

In this thesis we presented a comprehensive experimental analysis of footstep GRF-based person recognition using four different feature extractors, seven different normalizers, and five different classifiers. Our findings, though subject to the caveats in the previous section, demonstrated that with an appropriate choice of feature extractor, normalizer and classifier strong recognition performance can be achieved. That being said, we believe further improvements could be gained by both expanding the optimization of the recognition techniques studied in this thesis and testing promising alternative recognition techniques. In the following subsections we examine such potential improvements as they pertain to each of our biometric system components.

8.3.1 Feature Extraction

In section 8.2.2 we identified the choice of feature extractor as perhaps the most important factor in achieving strong GRF recognition results. For the purpose of this thesis all feature extractors were optimized using the KNN classifier and, as a result of our optimization, we may have formed a bias toward expectations of that classifier. It then stands to reason that we may be able to improve upon our recognition performance by simply re-running the feature space optimization for each of our different classifiers to obtain results unaffected from the KNN bias. Alternatively, shifting our focus toward other feature extraction techniques, there are several other techniques that have the potential to offer improved recognition results including: Partial Least Squares (PLS) [104], Kernel Principal Component Analysis (KPCA) [105], and Generalized Principal

Component Analysis (GPCA) [5]. Each of these techniques have roots in PCA, though two of them (PLS and GPCA) take a supervised extraction approach, applying a weighting based on class membership to derive their dimensionality reducing transformations, while the KPCA technique is simply the application of PCA over the kernel space, just as was demonstrated for our LDA classification technique in going from ULDA to KUDA.

8.3.2 Normalization

The options available to improve upon recognition performance via normalization improvements were more limited when compared with those available to feature extraction. Our work covered most of the normalizers recommended in statistics literature, and of all our examined normalizers only the LLSRDTW involved any parameter optimization, with the Sakeo-Chiba Band as a single optimization parameter. In our work this optimization was done using the KNN, again creating a bias for that particular classifier, therefore it is possible that redoing this optimization over other classifiers could potentially lead to an improvement in GRF recognition performance. Alternatively, though not quite normalization by definition, we may be able to achieve an improvement in our GRF recognition results by dropping the regression aspect of LLSRDTW and using the technique's sample alignment procedure as a standalone preprocessor; in this case we would be performing feature extraction directly on the center star aligned spaces rather than via derived warping functions. Another possibility might include using DTW and the center star algorithm to generate separate alignments

for each subject to capture inherent characteristics that might otherwise be missed in a more general modeling; again this would involve passing the aligned spaces directly into the feature extractor.

8.3.3 Classification

Classification offers far more choices for potentially improving upon recognition results than were available for the two preceding biometric system components. There are numerous classification techniques available for all sorts of specialized purposes; moreover, classifiers tend to have massive optimization spaces meaning whatever parameters are discovered in a standard parameter optimization are unlikely to be in their most optimal state. Classifier optimization in nearly all our classifiers was accomplished using brute force searches of arbitrarily chosen numeric intervals. Consequently, we believe improved performance could be achieved over most of the classifiers examined in this thesis via the incorporation of smarter parameter space optimization techniques. Another approach to improve upon current recognition performance might be to include a training sample rejection strategy, as was proposed in [32], or perhaps a fusing of different feature spaces for classification (also proposed in that same study). Alternatively, there are several other classification techniques that could potentially lead to improved recognition performance: the Hidden Markov Model (HMM) classifier [30], the Regularized Linear Discriminant Analysis (RLDA) classifier [92], and the Max-Margin Markov Network (M^3-net) classifier [106]. The HMM is a generative classification technique which was previously used for GRF recognition in [30] and is

often credited for its strength in recognizing temporal patterns. The RLDA classifier is an LDA variant that might be expected to achieve similar results when compared to our LDA classifier, but is said to be less prone to the overfitting problems that impacted our tested LDA variant. Finally, the M^3-net classifier is a leading edge classifier, which incorporates concepts from both the HMM and SVM classifiers to derive classification models; it is unlikely to have ever been tested for the purpose of GRF recognition, but, if it truly benefits from the properties of both SVM and HMM, it could potentially lead to a substantial improvement in recognition results.

8.4 Summary

In this chapter we performed a thorough analysis of the results obtained in the previous chapter. We began the chapter with a detailed look at the recognition performance achieved in chapter 7 and compared our findings with those of related GRF recognition studies. We continued on to discuss the feasibility of a deployment of our biometric system to a practical setting, identifying issues that were not within the scope of this thesis, yet would need to be addressed before attempting such a deployment. Finally, the chapter concluded with a look into some promising alternative recognition techniques together with ways in which we may be able to optimize our current recognition techniques for better performance going forward.

Key findings presented in this chapter supported our problem statement assertions. We discovered an approximately 50% decrease in recognition performance as a result of footwear variations between biometric system training and testing. Moreover, a 14-15% increase in recognition performance was observed with the use of LLSRDTW to normalize for stepping speed over our best performing holistic and wavelet feature spaces. Additionally, when comparing different feature spaces, our wavelet space produced the best results with an EER of about 2.6% averaged across all classifiers. And, when comparing different classifiers, we found the LDA classifier achieved the best overall performance, about 19% better than the next best classifier. Yet, to deploy this biometric system to a practical setting, there are issues that must be addressed, and, as discussed throughout section 8.2, we believe some of our recognition techniques may be undesirable options when computational efficiency is considered.

Chapter 9

Conclusion

The study of behavioural biometrics has revealed a number of powerful new person-distinguishing characteristics, some with the potential to be both less intrusive and more fraud-resistant when compared with other security mechanisms like physical biometrics. However, the complicated data samples that often accompany behavioural biometrics present the need for a new generation of processing and classification techniques; techniques able to identify key traits while not being significantly influenced by intra-person variability. In this thesis we performed a comprehensive analysis of such techniques and sources of variability with the intended purpose of performing person recognition via the behavioural biometric known as the footstep Ground Reaction Force (GRF), which is a form of Gait Biometric. Through our work we demonstrated two novel machine learning-based normalization techniques, supported two assertions relating to the effects of shoe type and stepping speed on GRF recognition performance, and compared a number of feature extractor, normalizer, and classifier configurations that had never before been cross examined with respect to GRF person recognition.

9.1 Contributions

This thesis made several significant contributions to both the study of footstep GRF-based person recognition and the wider field of machine learning. With respect to GRF-based person recognition the work presented here backed the assertions that intra-person

shoe type variations have a negative impact on recognition performance and that normalizing for stepping speed will have a positive impact on recognition performance. And with respect to machine learning, we presented a detailed theoretical overview of many existing machine learning techniques as well as two novel data preprocessing techniques that we developed for the purpose of our research, the Localized Least Squares Regression (LLSR) normalizer and the Localized Least Squares Regression with Dynamic Time Warping (LLSRDTW) normalizer.

Regarding our first GRF recognition assertion, to our knowledge no previous GRF-recognition study has performed a thorough analysis of the effect of shoe type on recognition performance. To address this research gap we divided our evaluation dataset into three different testing sets: one containing samples that always had the same shoe type as the training set, one containing only samples with a different shoe type from the training set, and one containing a mix of samples among which some were from the same shoe type as the training set while others differed. After applying five different classifiers to evaluate these subsets we discovered an average decrease in recognition performance of about 50% when the shoes that were tested differed from those on which the classifier was trained.

When evaluating our second assertion once again we could not find any previous supporting GRF-recognition studies; nor could we find much information in the general statistical/machine learning literature demonstrating effective preprocessing techniques to account for temporal variation between data samples. In our work we postulated that

models could be derived to map the relationship between step duration and amplitude at various localities within the GRF signature. To prove this we developed a new normalization technique (LLSR); this technique generated its models via first aligning the training data at each data point in relation to the total step duration and then acquiring the resulting least squares regression relationships. We later used the Dynamic Time Warping (DTW) technique to better align samples prior to acquiring the regression models, giving us another normalization technique, which we refer to as LLSRDTW. To perform the actual normalization on our dataset we then assigned each sample a common step duration and used the derived models to adjust the sample GRF signatures accordingly. In our evaluation experiments we compared the non-normalized GRF recognition performance with the performance achieved using our two new normalization techniques as well as a third technique known as Linear Time Normalization (LTN). What we found was that each step duration normalizer resulted in an improvement in recognition performance for almost every feature space (with the exception of the geometric feature space). The largest improvement came from our new LLSRDTW normalizer, with a 14-15% increase in GRF recognition performance compared with its non-normalized equivalent over our two best feature spaces (the holistic and wavelet spaces).

Aside from supporting the two aforementioned assertions, this thesis also made several other contributions to the study of GRF recognition. To our knowledge our examined sample space was larger than the sample spaces of all previous GRF recognition research, with about 12800 points per sample covering 8 different GRF signals. Furthermore, this

research, for the first time, performed GRF recognition using the Least Square Probabilistic Classification (LSPC) classifier. It was also the first to compare the wavelet packet feature extraction technique to the spectral and holistic techniques, the holistic feature extraction technique to the spectral, and the Multi-layer Perceptron (MLP) classifier to the Linear Discriminant Analysis (LDA) classifier. Through this work we were able to demonstrate the wavelet packet feature extractor as being superior to all other tested feature extractors with an average EER of about 2.6% and best EER of about 1.6% in our evaluation experiments. Moreover, we found the LDA classifier performed the best in our evaluation tests, about 19% better than the next best classifier on average; however, performance concerns were raised with regards to its applicability in a practical setting.

9.2 Future Work

The work presented in this thesis tied together the analysis of a number of powerful preprocessing and classification techniques, providing a foundation upon which future work can be built. While the objectives of this thesis were achievable over a relatively limited dataset of only 10 different subjects, future work would benefit greatly from the use of a more realistic dataset. Ideally the future dataset would contain over 100 different subjects with varying footwear and with steps collected at varying walking speeds; during testing, this dataset should then be divided into enrolled subjects and previously unseen imposter subjects. In analyzing such a dataset we would expect to get a more accurate measure of performance that could be expected in a real world setting. We also expect

that many of our machine learning technique preprocessors and classifiers would benefit from the inclusion of more samples during training. However, a larger dataset may expose some of the potential flaws in our better performing biometric system configurations, namely computational inefficiencies. In future work we would like to perform a thorough analysis of the computational growth rate introduced by each configuration to achieve a greater understanding of the underlying performance-efficiency trade-offs. We also believe the study of footstep GRF recognition would benefit greatly from a similar assessment of the impact of training/testing feet (left vs right) on recognition performance.

In terms of improving the recognition performance established in this thesis, our experimental analysis made several suggestions. Many of our existing classifiers likely have room for improvement through their direct use in the optimization of our preprocessors; in our work all preprocessors were optimized exclusively using K Nearest Neighbour (KNN) classifier. Additionally, we believe any future analysis would benefit from the inclusion of alternative feature extractors and classifiers. New classifiers to be tested might include the Hidden Markov Model (HMM) classifier, the Max-Margin Markov Network (M^3-net) classifier, and/or different variants of the classifiers examined in this thesis, such as regularized-LDA (RLDA). New feature extractors to be tested might include the Partial Least Squares (PLS) feature extractor, the Kernel Principal Component Analysis (KPCA) feature extractor and/or the Generalized PCA (GPCA) feature extractor. Another interesting idea introduced in our experimental analysis involves using DTW and the center star approximation algorithm as a stand-alone data

preprocessor to be run prior to any feature extraction. Furthermore, we believe that our GRF recognition performance could be further improved by using strategies to both reject undesirable samples from training and to fuse results obtained over multiple different feature spaces.

Finally, in addition to building upon the research presented here for the purpose of furthering future understanding of the footstep GRF recognition domain, much of the work presented here also could be applicable to a far wider field of classification problems. This is particularly true for the machine learning-based techniques that were used, which we feel could be applied to a number of other domains with little-to-no domain knowledge required. In future work we would like to take the technologies presented here, together with some of the suggested potential improvements, and apply them to other domains such as speech recognition. It is our hope that this thesis can become something of a template when faced with a classification problem in an unfamiliar domain.

Bibliography

[1] C. Roberts, "Biometric attack vectors and defenses," *Computers & Security 26,* pp. 14-25, 2007.

[2] R. V. Yampolskiy, "Behavioral Modeling: an Overview," *American Journal of Applied Sciences 5 (5),* pp. 496-503, 2008.

[3] R. V. Rodríguez, N. W. D. Evans, R. P. Lewis, B. Fauve and J. S. D. Mason, "An experimental study on the feasibility of footsteps as a biometric," in *15th European Singal Processing Conference (EUSIPCO 2007)*, Poznan, Poland, 2007.

[4] R. V. Rodríguez, J. S. D. Mason and N. W. D. Evans, "Footstep Recognition for a Smart Home Environment," *International Journal of Smart Home,* vol. 2, no. 2, pp. 95-110, 2008.

[5] P. C. Cattin, "Biometric Authentication System Using Human Gait," Swiss Federal Institute of Technology, Zurich, Switzerland, 2002.

[6] A. J. Taylor, H. B. Menz and A.-M. Keenan, "The influence of walking speed on plantar pressure measurements using the two-step gait initiation protocol," *The Foot 14,* pp. 49-55, 2004.

[7] S. P. Moustakidis, J. B. Theocharis and G. Giakas, "Subject recognition based on ground reaction force measurements of gait signals," *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 38, no. 6, pp. 1476-1485, 2008.

[8] M. S. Nixon and J. N. Carter, "Automatic Recognition by Gait," *Preceedings of the IEEE,* vol. 94, no. 11, pp. 2013-2024, 2006.

[9] D. Gafurov, "A Survey of Biometric Gait Recognition: Approaches, Security and Challenges," in *Annual Norwegian Computer Science Conference*, Bergen, Norway, 2007.

[10] M. O. Derawi, D. Gafurov and P. Bours, "Towards Continuous Authentication Based on Gait Using Wearable Motion Recording Sensors," in *In Continuous Authentication Using Biometrics: Data, Models, and Metrics, I. Traore & A. Ahmed (Eds.)*, IGI Global, 2012, pp. 170-192.

[11] M. Nixon, "Gait Biometrics," *Biometric Technology Today,* vol. 16, no. 7-8, pp. 8-9, 2008.

[12] M. H. Cheng, M. F. Ho and C. L. Huang, "Gait analysis for human identification through manifold learning and HMM," *Pattern Recognition,* vol. 41, no. 8, pp. 2541-2553, 2008.

[13] L. Wang, T. Tan, H. Ning and W. Hu, "Silhouette Analysis-Based Gait Recognition for Human Identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 25, no. 12, pp. 1505-1517, 2003.

[14] N. V. Boulgouris, P. N. Konstantinos and D. Hatzinakos, "Gait recognition using linear time normalization," *Pattern Recognition 39,* pp. 969-979, 2006.

[15] J. Lu and E. Zhang, "Gait Recognition for Human Identification based on ICA and Fuzzy SVM Through Multiple Views Fusion," *Pattern Recognition Letters 28,* pp. 2401-2411, 2007.

[16] Z. Liu and S. Sarkar, "Improved Gait Recognition by Gait Dynamics Normalization," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 28, no. 6, pp. 863-876, 2006.

[17] I. Venkat and P. De Wilde, "Robust gait recognition by learning and exploiting sub-gait characteristics," *Int. J. Comput. Vis 91(1),* pp. 7-23, 2011.

[18] I. Bouchrika and M. S. Nixon, "Model-Based Feature Extraction for Gait Analysis and Recognition," in *Mirage: Computer Vision / Computer Graphics Collaboration Techniques and Applications,* Inria Rocquencourt, France, 2007.

[19] I. Bouchrika and M. S. Nixon, "Exploratory Factor Analysis of Gait Recognition," in *FG '08. 8th IEEE International Conference on Automatic Face & Gesture Recognition*, Amsterdam, Netherlands, 2008.

[20] X. Zhou and B. Bhanu, "Feature Fusion of Face and Gait for Human Recognition at a Distance in Video," in *Proceedings of IEEE International Conference Pattern Recognition*, Hong Kong, China, 2006.

[21] A. Itai and H. Yasukawa, "Person Identification Using Footstep Based on Wavelets," in *International Symposium on Intelligent Signal Processing and Communication Systems,* Totoori, Japan, 2006.

[22] S. J. M. Bamberg, A. Y. Benbasat, D. M. Scarborough, D. E. Krebs and J. A. Paradiso, "Gait Analysis Using a Shoe Integrated Wireless Sensor System," *IEEE Trans. Inf. Technol. Biomed,* vol. 12, no. 4, pp. 413-423, 2008.

[23] D. Gafurov, K. Helkala and S. Torkjel, "Biometric Gait Authentication Using Accelerometer Sensor," *Journal of Computers,* vol. 1, no. 7, pp. 51-58, 2006.

[24] B. Huang, M. Chen, W. Ye and Y. Xu, "Intelligent Shoes for Human Identification," in *IEEE International Conference on Robotics and Biomimetics*, Kunming, China, 2006.

[25] B. Huang, M. Chen, P. Huang and Y. Xu, "Gait Modeling for Human Identification," in *IEEE International Conference on Robotics and Automation*, Roma, Italy, 2007.

[26] S. Spranger and D. Zazula, "Gait Identification Using Cumulants of Accelerometer Data," in *2nd WSEAS International Conference on Sensors, and Signals and Visualization, Imaging and Simulation and Materials Science*, Stevens Point, Wisconsin, USA, 2009.

[27] M. N. Fitzgerald, "Human Identification via Gait Recognition Using Accelerometer Gyro Force," 2009. [Online]. Available: http://www.cs.yale.edu/homes/mfn3/pub/mfn_gait_id.pdf. [Accessed Mar 2012].

[28] M. O. Derawi, P. Bours and K. Holien, "Improved Cycle Detection for Accelerometer Based Gait Authentication," in *Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*, Darmstadt, Germany, 2010.

[29] E. S. Sazonov, T. Bumpus, S. Zeigler and S. Marocco, "Classification of Plantar Pressure and Heel Acceleration Patterns Using Neural Networks," in *IEEE International Joint Conference on Neural Networks (Vol. 5)*, Montreal, Canada, 2005.

[30] M. D. Addlesee, A. H. Jones, F. Livesey and F. S. Samaria, "The ORL active floor [sensor system]," *IEEE Personal Communications,* vol. 4, no. 5, pp. 35-41, 1997.

[31] R. J. Orr and G. D. Abowd, "The Smart Floor: A Mechanism for Natural User Identification and Tracking," in *CHI '00, Conference on Human Factors in Computer Systems*, The Hague, Netherlands, 2000.

[32] J. Suutala and J. Röning, "Methods for person identification on a pressure-sensitive floor: Experiments with multiple classifiers and reject option," *Information Fusion Journal, Special Issue on Applications of Ensemble Methods 9,* pp. 21-40, 2008.

[33] H. Ye, S. Kobashi, Y. Hata, K. Taniguchi and K. Asari, "Biometric System by Foot Pressure Change Based on Neural Network," in *39th International Symposium on Multiple-Valued Logic*, Naha, Okinawa, Japan, 2009.

[34] J. Suutala, K. Fujinami and J. Röning, "Gaussian Process Person Identifier Based on Simple Floor Sensors," in *Smart Sensing and Context Third European Conference, EuroSSC*, Zurich, Switzerland, 2008.

[35] J. Yun, "User identification using gait patterns on UbiFloorII," *Sensors 11*, pp. 2611-2639, 2011.

[36] "Swansea Footstep Recognition Dataset," Swansea University, [Online]. Available: http://eeswan.swan.ac.uk. [Accessed 03 Sept 2014].

[37] "Plantiga," [Online]. Available: http://www.plantiga.com. [Accessed 02 Sept 2014].

[38] "Kistler force plate formulae," [Online]. Available: http://isbweb.org/software/movanal/vaughan/kistler.pdf. [Accessed 18 Mar 2012].

[39] A. Mostayed, S. Kim, M. M. G. Mazumder and S. J. Park, "Foot Step Based Person Identification Using Histogram Similarity and Wavelet Decomposition," in *2008 International Conference on Information Security and Assurance*, Busan, Korea, 2008.

[40] G. Bouchard and B. Triggs, "The Trade-Off Between Generative and Discriminative Classifiers," in *COMPSTAT 2004*, Prague, Czech Republic, 2004.

[41] A. Y. Ng and M. I. Jordan, "On Discriminative vs. Generative classifiers: A comparison of logistic regression and naive Bayes," in *Advances in Neural Information Processing Systems 14 (NIPS 2001)*, Vancouver, Canada, 2001.

[42] M. Sugiyama, "Superfast-Trainable Multi-Class Probabilistic Classifier by Least-Squares Posterior Fitting," *IEICE Transaction on Information and Systems*, Vols. E93-D, no. 10, pp. 2690-2701, 2010.

[43] F. Zhang, "Cross-Validation and Regression Analysis in High Dimensional Sparse Linear Models," Standford University, Standford, California, USA, 2011.

[44] D. A. Winter, A. E. Patla, J. S. Frank and S. E. Walt, "Biomechanical Walking Pattern Changes in the Fit and Healthy Elderly," *Physical Therapy Jouranl of American Physical Therapy Association*, vol. 70, pp. 340-347, 1990.

[45] J. L. Castellanos, G. Susan and V. Guerra, "The triangle method for finding the corner of the L-curve," *Applied Numerical Mathematics 43*, pp. 359-373, 2002.

[46] I. T. Jolliffee, "Introduction," in *Principal Component Analysis, Springer Series in Statistics*, Springer Verlag, 2002, pp. 1-9.

[47] L. I. Smith, "A tutorial on principal component analysis," Cornell University, Cornell, USA, 2002.

[48] C. Souza, "Principal Component Analysis in C#," Sept 2009. [Online]. Available: http://crsouza.blogspot.ca/2009/09/principal-component-analysis-in-c.html. [Accessed 3 Sept 2014].

[49] I. T. Jolliffee, "The Singular Value Decomposition," in *Principal Component Analysis, Springer Series in Statistics*, Springer Verlag, 2002, pp. 44-46.

[50] "13.4 Power Spectrum Estimation Using the FFT," in *Numeric Recipes in C*, Cambridge University Press, 1993, pp. 549-558.

[51] P. M. Bentley and J. T. E. McDonnell, "Wavelet transformations: an introduction," *Electronics & Communication Engineering Journal*, pp. 175-186, 1994.

[52] D. Li, W. Pedrycz and N. J. Pizzi, "Fuzzy Wavelet Packet Based Feature Extraction Method and Its Application to Biomedical Signal Classification," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 6, pp. 1132-1139, 2005.

[53] C. Scheiblich, "JWave," [Online]. Available: https://code.google.com/p/jwave/. [Accessed 3 Sept 2014].

[54] X. Wang, L. M. Wang and Q. Yu, "A Comparative Study of Encoding, Pooling, and Normalization Methods for Action Recognition," in *ACCV'12 Proceedings of the 11th Asian conference on Computer Vision Volume Part III*, Daejeon, Korea, 2012.

[55] D. Nistér and H. Stewénius, "Scalable Recognition with a Vocabulary Tree," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, New York, NY, USA, 2006.

[56] Q. Zhu, S. Avidan, M.-C. Yeh and K.-T. Cheng, "Fast Human Detection Using a Cascade of Histograms of Oriented Gradients," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, New York, NY, USA, 2006.

[57] Z. Cao, Q. Yin, X. Tang and J. Sun, "Face Recognition with Learning-based Descriptor," in *2010 IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco, CA, USA, 2010.

[58] D. Byrd, S. Lee and R. Compos-Astorkiza, "Phrase boundary effects on the temporal kinematics of sequential tongue tip consonants," *J. Acoust. Soc. Am. 123 (6)*, pp. 4456-4465, 2008.

[59] C. Barras and J.-L. Gauyain, "Feature and Score Normalization for Speaker Verification of Cellular Data," *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '03) (Vol. 2),* pp. 49-52, 2003.

[60] G. Quinn and M. Keough, "Analysis of Covariance," in *Experimental Design and Data Analysis for Biologists*, Cambridge University Press, 2002, pp. 339-358.

[61] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing,* vol. 26, no. 1, pp. 43-49, 1978.

[62] K. Wang and T. Gasser, "Alignment of Curves by Dynamic Time Warping," *The Annals of Statistics,* vol. 25, no. 3, pp. 1251-1276, 1997.

[63] P. Sanguansat, "Multiple Multidimensional Sequence Alignment Using Generalized Dynamic Time Warping," *WSEAS Transactions on Mathematics,* vol. 11, no. 8, pp. 668-678, 2012.

[64] D. Gusfield, "Efficient methods for multiple sequence alignment with guaranteed error bounds," *Bulletin of Mathematical Biology,* vol. 55, no. 1, pp. 141-154, 1993.

[65] T. N. Phyu, "Survey of Classification Techniques in Data Mining," in *International MultiConference of Engineers and Computer Scientists 2009 Vol I IMECS 2009,* Hong Kong, China, 2009.

[66] D. Wettschereck, "A Study of Distance-Based Machine Learning Algorithms," Oregon State University, Corvallis, OR, USA, 1994.

[67] J. M. Keller, M. R. Gray and J. A. Givens, Jr, "A Fuzzy K-Nearest Neighbor Algorithm," *IEEE transactions on systems, man, and cybernetics,* Vols. SMC-15, no. 4, pp. 580-585, 1985.

[68] R. Rojas, "The Biological Paradigm," in *Neural Networks,*, Berlin, Germany, Springer-Verlag, 1996, pp. 3-26.

[69] Y. LeCun and Y. Bengio, "Pattern Recognition," in *The Handbook of Brain Theory and Neural Networks,* A Bradford Book, 1995, pp. 864-868.

[70] J. C. Principe, N. R. Euliano and W. C. Lefebvre, "Multilayer Perceptron," in *Neural and Adaptive Systems: Fundamentals Through Simulation*, Wiley , 1999, pp. 100-172.

[71] M. Hajek, "Models of a neuron," in *Neural Networks*, Durban, South Africa, University of KwaZulu-Natal, 2005, pp. 9-10.

[72] M. Fernández-Redondo and C. Hernández-Espinosa, "Weight Initialization Methods for Multilayer Feedforward," in *European Symposium on Artificial Neural Networks*, Bruges, Belgium, 2001.

[73] R. Rojas, "The Backpropagation Algorithm," in *Neural Networks*, Berlin, Germany, Springer-Verlag, 1996, pp. 149-180.

[74] J. Heaton, "Encog Machine Learning Framework," Heaton Research, 2014. [Online]. Available: http://www.heatonresearch.com/encog. [Accessed 3 Sept 2014].

[75] J. Rynkiewicz, "General bound of overfitting for MLP regression models," *Neurocomputing 90*, pp. 106-110, 2012.

[76] S. Lawrence, C. L. Giles and A. C. Tsoi, "Lessons in Neural Network Training: Overfitting May be Harder than Expected," in *Fourtheenth National Conference on Artificial Intelligence*, Manlo Park, CA, USA, 1997.

[77] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Mining and Knowledge Discovery 2*, pp. 121-167, 1998.

[78] J. Milgram, M. Cheriet and R. Sabourin, ""One Against One" or "One Against All": Which One is Better for Handwritting Recognition with SVMs?," in *Tenth International Workshop on Frontiers in Handwriting Recognition*, La Baule, France, 2006.

[79] A. Agapitos, O. Michael and A. Brabazon, "Maximum Margin Decision Surfaces Increased Generalisation in Evolutionary Decision Tree Learning," in *Genetic Programming*, Springer Berlin Heidelberg, 2011, pp. 61-72.

[80] P.-H. Chen, C.-J. Lin and B. Schölkopf, "A tutorial on v-support vector machines," *Appl Stochastic Models Bus Ind, 21*, pp. 111-136, 2005.

[81] L. Bottou and C.-J. Lin, "Support Vector Machine Solvers," in *Large-Scale Kernel Machines*, Cambridge, MA, USA, MIT Press, 2007, pp. 1-27.

[82] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," 2001. [Online]. Available: http://www.csie.ntu.edu.tw/~cjlin/papers/libsvm.pdf. [Accessed 20 Nov 2013].

[83] H.-T. Lin, C.-J. Lin and R. C. Weng, "A Note on Platt's Probabilistic Outputs for Support Vector Machines," *Machine Learning*, vol. 68, no. 3, pp. 267-276, 2007.

[84] T.-F. Wu, C.-J. Lin and R. C. Weng, "Probability Estimates for Multi-class Classification by Pairwise Coupling," *Journal of Machine Learning Research 5*, pp. 975-1005, 2004.

[85] T. Hastie, R. Tibshirani and A. Buja, "Flexible Discriminant Analysis by Optimal Scoring," *Journal of the American Statistical Association,* vol. 89, no. 428, pp. 1255-1270, 1994.

[86] R. Huang, Q. Liu, H. Lu and S. Ma, "Solving the Small Sample Size Problem of LDA," in *16th International Conference on Pattern Recognition*, Quebec City, Canada, 2002.

[87] J. Ye, "Characterization of a Family of Algorithms for Generalized Discriminant Analysis on Undersampled Problems," *Journal of Machine Learning Research 6*, pp. 483-502, 2005.

[88] L. Wang, L. Bo and L. Jiao, "Kernel Uncorrelated Discriminant Analysis for Radar Target Recognition," in *Neural Information Processing*, Springer Berlin Heidelberg, 2006, pp. 404-411.

[89] S. Srivastava, M. R. Gupta and B. A. Frigyik, "Bayesian Qaudratic Discriminant Analysis," *Journal of Machine Learning Research 8*, pp. 1277-1305, 2007.

[90] T. Balachander, R. Kothari and H. Cualing, "An Empirical Comparison of Dimensionality Reduction Techniques for Pattern Classification," in *Artificial Neural Networks - ICANN'97*, Springer Berlin Heidelberg, 1997, pp. 589-594.

[91] E. Hidayat, N. A. Fajrian, A. K. Muda, C. Y. Huoy and S. Ahmad, "A Comparative Study of Feature Extraction Using PCA and LDA for Face Recognition," in *7th International Conference on Information Assurance and Security (IAS)*, Melaka, Malaysia, 2011.

[92] J. H. Friedman, "Regularized Discriminant Analysis," *Journal of the American Statistical Association,* vol. 84, no. 405, pp. 165-175, 1989.

[93] W. Liu, Y. Wang, S. Z. Li and T. Tan, "Null Space Approach of Fisher Discriminant Analysis for Face Recognition," in *Biometric Authentication*, Springer Berlin Heidelberg, 2004, pp. 32-44.

[94] T. Hastie, A. Buja and R. Tibshirani, "Penalized Discriminant Analysis," *The Annals of Statistics,* vol. 23, no. 1, pp. 73-102, 1995.

[95] "What Is the Generalized Inverse of a Matrix?," [Online]. Available: http://artsci.wustl.edu/~jgill/papers/ginv.pdf. [Accessed 9 May 2014].

[96] K. Liu, Y.-Q. Cheng and J.-Y. Yang, "A Generalized Optimal Set of Discriminant Vectors," *Pattern Recognition*, vol. 25, no. 7, pp. 731-739, 1992.

[97] S. J. Orfanidis, "SVD, PCA, KLT, CCA, and All That," Rutger University, 2007.

[98] C. H. Park and H. Park, "Nonlinear Discriminant Analysis using Kernel Functions and the Generalized Singular Value Decomposition," *SIAM Journal on Matrix Analysis and Applications 27(1),* pp. 87-102, 2005.

[99] P. Selormey, "DotNetMatrix: Simple Matrix Library for .NET," 12 Jan 2004. [Online]. Available: http://www.codeproject.com/Articles/5835/DotNetMatrix-Simple-Matrix-Library-for-NET. [Accessed 3 Sept 2014].

[100] J. Hicklin, C. Moler, P. Webb, F. R. Boisvert, B. Miller, R. Pozo and K. Remington, "Jama: A Java Matrix Package," NIST, 2012. [Online]. Available: http://math.nist.gov/javanumerics/jama/. [Accessed 3 Sept 2014].

[101] J. Ye, T. Xiong, Q. Li, R. Janardan, J. Bi, V. Cherkassky and C. Kambhamettu, "Efficient Model Selection for Regularized Linear Discriminant Analysis," in *15th ACM International Conference on Information and Knowledge Management*, New York, NY, USA, 2006.

[102] H. Hachiya, M. Sugiyama and N. Ueda, "Importance-Weighted Least Squares Probabilistic Classifier for Covariate Shift Adaption with Application to Human Activity Recognition," *Neurocomputing*, vol. 80, pp. 93-101, 2012.

[103] D. Kozen and M. Timme, "Idefinite Summation and the Kronecker Delta," 2007. [Online]. Available: http://dspace.library.cornell.edu/bitstream/1813/8352/2/Kronecker.pdf. [Accessed 14 June 2014].

[104] W. R. Schwartz, A. Kembhavi, D. Harwood and L. S. Davis, "Human Detection Using Partial Least Square Analysis," in *2009 IEEE 12th Internation Conference on Computer Vision*, Kyoto, Japan, 2009.

[105] B. Schölkopf, A. Smola and K.-R. Müller, "Kernel Principal Component Analysis," in *Artificial Neural Networks - ICANN'97*, Springer Berlin Heidelberg, 1997, pp. 583-588.

[106] S.-L. Julien, "Combining SVM with graphical models for supervised classification: an introduction to Max-Margin Markov Networks," University of California, Berkeley, CA, USA, 2003.