TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

MACHINE LEARNING FOR THE CLASSIFICATION OF ATRIAL FIBRILLATION UTILIZING SEISMO- AND GYROCARDIOGRAM

Saeed Mehrang

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. F OSA – TOM. 30 | TECHNICA – INFORMATICA | TURKU 2023

TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

MACHINE LEARNING FOR THE CLASSIFICATION OF ATRIAL FIBRILLATION UTILIZING SEISMO- AND GYROCARDIOGRAM

Saeed Mehrang

TURUN YLIOPISTON JULKAISUJA – ANNALES UNIVERSITATIS TURKUENSIS
SARJA – SER. F OSA – TOM. 30 | TECHNICA – INFORMATICA| TURKU 2023

University of Turku

Faculty of Technology
Department of Computing
Information and Communication Technology
Doctoral Programme in Technology (DPT)

Supervised by

Prof. Pasi Liljeberg
University of Turku

Assoc. Prof. Antti Airola
University of Turku

Adjunct. Prof. Mikko Pänkäälä
University of Turku

Reviewed by

Prof. Omer Inan
Georgia Institute of Technology

Assoc. Prof. Kouhyar Tavakolian
University of North Dakota

Opponent

Assoc. Prof. Samuel Emil Schmidt
Aalborg University

ISBN 978-951-29-9542-4 (PRINT)
ISBN 978-951-29-9543-1 (PDF)
ISSN 2736-9390 (PRINT)
ISSN 2736-9684 (ONLINE)
Painosalama, Turku, Finland 2023

I dedicate this thesis to my late father and grandfather
Shahram and Yosuf Mehrang

UNIVERSITY OF TURKU
Faculty of Technology
Department of Computing
Information and Communication Technology
MEHRANG, SAEED: Machine Learning for the Classification of Atrial Fibrillation utilizing Seismo- and Gyrocardiogram
Doctoral dissertation, 138 pp.
Doctoral Programme in Technology (DPT)
December 2023

ABSTRACT

A significant number of deaths worldwide are attributed to cardiovascular diseases (CVDs), accounting for approximately one-third of the total mortality in 2019, with an estimated 18 million deaths. The prevalence of CVDs has risen due to the increasing elderly population and improved life expectancy. Consequently, there is an escalating demand for higher-quality healthcare services. Technological advancements, particularly the use of wearable devices for remote patient monitoring, have significantly improved the diagnosis, treatment, and monitoring of CVDs.

Atrial fibrillation (AFib), an arrhythmia associated with severe complications and potential fatality, necessitates prolonged monitoring of heart activity for accurate diagnosis and severity assessment. Remote heart monitoring, facilitated by ECG Holter monitors, has become a popular approach in many cardiology clinics. However, in the absence of an ECG Holter monitor, other remote and widely available technologies can prove valuable. The seismo- and gyrocardiogram signals (SCG and GCG) provide information about the mechanical function of the heart, enabling AFib monitoring within or outside clinical settings. SCG and GCG signals can be conveniently recorded using smartphones, which are affordable and ubiquitous in most countries.

This doctoral thesis investigates the utilization of signal processing, feature engineering, and supervised machine learning techniques to classify AFib using short SCG and GCG measurements captured by smartphones. Multiple machine learning pipelines are examined, each designed to address specific objectives. The first objective (O1) involves evaluating the performance of supervised machine learning classifiers in detecting AFib using measurements conducted by physicians in a clinical setting. The second objective (O2) is similar to O1, but this time utilizing measurements taken by patients themselves. The third objective (03) explores the performance of machine learning classifiers in detecting acute decompensated heart failure (ADHF) using the same measurements as O1, which were primarily collected for AFib detection. Lastly, the fourth objective (O4) delves into the application of deep neural networks for automated feature learning and classification of AFib.

These investigations have shown that AFib detection is achievable by capturing a joint SCG and GCG recording and applying machine learning methods, yielding satisfactory performance outcomes. The primary focus of the examined approaches encompassed (1) feature engineering coupled with supervised classification, and (2)

automated end-to-end feature learning and classification using deep convolutional-recurrent neural networks.

The key finding from these studies is that SCG and GCG signals reliably capture the heart's beating pattern, irrespective of the operator. This allows for the detection of irregular rhythm patterns, making this technology suitable for monitoring AFib episodes outside of hospital settings as a remote monitoring solution for individuals suspected to have AFib. This thesis demonstrates the potential of smartphone-based AFib detection using built-in inertial sensors. Notably, a short recording duration of 10 to 60 seconds yields clinically relevant results. However, it is important to recognize that the results for ADHF did not match the state-of-the-art achievements due to the limited availability of ADHF data combined with arrhythmias as well as the lack of a cardiopulmonary exercise test in the measurement setting.

Finally, it is important to recognize that SCG and GCG are not intended to replace clinical ECG measurements or long-term ambulatory Holter ECG recordings. Instead, within the scope of our current understanding, they should be regarded as complementary and supplementary technologies for cardiovascular monitoring.

KEYWORDS: Atrial fibrillation, cardiovascular diseases, machine learning, signal processing

TURUN YLIOPISTO
Teknillinen tiedekunta
Tietotekniikan laitos
Tieto- ja viestintatekniikka
MEHRANG, SAEED: Machine Learning for the Classification of Atrial Fibrillation utilizing Seismo- and Gyrocardiogram
Väitöskirja, 138 s.
Teknologian tohtoriohjelma (DPT)
Joulukuu 2023

TIIVISTELMÄ

Sydän- ja verisuonitaudit (CVD) aiheuttavat maailmanlaajuisesti merkittävästi kuolleisuutta ja sairastuvuutta. On arvioitu, että vuonna 2019 noin 18 miljoonaa kuolemantapausta johtui sydän- ja verisuonisairauksista, mikä on noin kolmasosa kaikista kuolemista maailmanlaajuisesti. Samalla kun iäkkäiden ihmisten määrä on tasaisesti kasvanut viime vuosikymmeninä elinajanodotteen pidentyessä, vastaavasti myös sydän- ja verisuonitautien esiintyvyys on lisääntynyt. Tämän seurauksena myös laadukkaampien terveydenhuoltopalvelujen tarve on lisääntynyt. Samaan aikaan sydän- ja verisuonitautien diagnosointi, hoito ja seuranta ovat hyötyneet suuresti teknologisesta edistyksestä, erityisesti puettavien laitteiden mahdollistamasta potilaiden etäseurannasta.

Eteisvärinä on rytmihäiriö, joka tyypillisesti aiheuttaa komplikaatioita, jotka voivat johtaa potilaan tilan heikkenemiseen tai jopa kuolemaan. Eteisvärinän diagnosointi ja vakavuuden tunnistaminen edellyttävät seurantatietoja sydämen toiminnasta päivien tai jopa viikkojen ajalta. Tämä on nykyään mahdollista sydämen etäseurantamenetelmien kehittymisen myötä. EKG Holter-tutkimuslaite on suosituin teknologia tähän tarkoitukseen. EKG Holterin avulla monet kardiologian klinikat pystyvät seuraamaan potilaiden sydämen toimintaa useiden päivien tai jopa viikkojen ajan. Muidenkin etä- ja jokapaikanteknologioiden käyttö voi myös olla hyödyllistä, etenkin jos EKG Holteria ei ole saatavilla. Seismo- ja gyrokardiogrammisignaalit (SCG ja GCG) antavat tietoa sydämen mekaanisesta toiminnasta, jota voidaan käyttää eteisvärinän seurantaan kliinisessä ympäristössä tai sen ulkopuolella. SCG ja GCG voidaan tallentaa älypuhelimilla, jollaisia on laajalti edullisesti saatavilla useimmissa maissa.

Tässä työssä tutkimme, kuinka signaalinkäsittelyä, piirteidensuunnittelua ja ohjattuja koneoppimismenetelmiä voidaan hyödyntää eteisvärinän luokittelussa älypuhelimilla tallennetuista lyhyistä SCG- ja GCG-mittauksista.

Tässä opinnäytetyössä on tutkittu useita koneoppimisjärjestelmiä, jotka kaikki on suunniteltu tiettyjä tavoitteita varten. Ensimmäinen tavoite (O1) oli arvioida ohjattujen koneoppimisluokittajien suorituskykyä eteisvärinän tunnistamiseksi perustuen lääkärien kliinisessä ympäristössä suorittamiin mittauksiin. Toinen tavoite (O2) oli muuten vastaava kuin O1, mutta tunnistus perustui potilaiden itsensä tekemiin mittauksiin. Kolmantena tavoitteena (O3) tutkimme koneoppimismenetelmien tarkkuutta akuutin dekompensoituneen sydämen vajaatoiminnan tunnistamiseksi(ADHF)

perustuen täsmällisiin mittauksiin, jotka kerättiin ensisijaisesti eteisvärinän havaitsemiseksi. Viimeisenä tavoitteena (O4) tutkimme mahdollisuutta käyttää syviä hermoverkkoja automatisoituun piirteidenoppimisen ja eteisvärinän tunnistamiseen.

Tutkimuksessa havaitsimme, että eteisvärinä voidaan tunnistaa tyydyttävällä tarkkuudella koneoppimisen avulla perustuen lyhyeen yhteiseen SCG- ja GCG-tallenteeseen. Ensisijaisesti tutkimuksessa käytetyt metodit keskittyivät (1) käsin valittuihin piirteisiin yhdistettynä tavalliseen luokittelijaan ja (2) automatisoituun päästä päähän -piirteiden oppimiseen ja syvään toistuvaan konvoluutiohermoverkkoon.

Tärkeimmät löydökset näistä tutkimuksista olivat, että SCG ja GCG signaalien avulla voidaan tallentaa sydämen lyöntimalli, riippumatta tallententajasta. Tämä mahdollistaa epäsäännöllisten rytmien havaitsemisen tällä teknologialla etänä myös sairaalaympäristön ulkopuolella, mahdollistaen rytmihäiriöiden havaitsemisen. Tämä opinnäytetyö demonstroi potentiaalia havaita rytmihäiriöt käyttäen vain matkapuhelinpohjaista detektoria käyttäen niiden sisäisiä sensoreita. Esimerkiksi jo 10-60 sekunnin tallennus antaa kliinisesti merkittäviä tuloksia. On kuitenkin tärkeää huomata, että tulokset sydämen vajaatoiminnan havaitsemiseksi eivät vastaa uusimpia tutkimuksen saavutuksia, johtuen datan heikosta saatavuudesta sydämen vajaatoiminnasta ja kardiovaskulaarisen kuntotestin puuttumisestta mittaustilanteessa.

Lopuksi on tärkeää huomata, että SCG ja GCG eivät korvaa kliinisesä ympäristössä toteutettua ECG mittausta tai pitkän tähtäimen Holter ECG tallennusta, vaan nämä tekniikat pitäisi nähdä täydentävinä rinnakkaisina keinoina sydän ja verisuonisairausten seurannassa.

ASIASANAT: Eteisvärinä, sydän- ja verisuonisairaudet, koneoppiminen, signaalinkäsittely

Acknowledgements

Firstly, I would like to thank my supervisors Prof. Pasi Liljeberg, Dr. Mikko Pänkäälä, and Assistant Prof. Antti Airola for their continuous support of my doctoral studies.

I would like to thank my reviewers, Prof. Omer Inan and Assoc. Prof. Kouhyar Tavakolian, for their invaluable enlightening comments. Moreover, I thank Assoc. Prof. Samuel Emil Schmidt, who kindly agreed to serve as the opponent of my doctoral defence.

In addition, I would like to express my appreciation to all of my collaborators and co-authors Dr. Mojtaba Jafari Tadi, Prof. Timo Knuutila, Tero Koivisto, Dr. Olli Lahdenoja, Dr. Matti Kaisti, Tero Hurnanen, Prof. Juhani Airaksinen, Dr. Tuomas Kiviniemi, Dr. Samuli Jaakkola, Dr. Jussi Jaakkola, and Tuija Vasankari.

My deepest gratitude is extended to my whole family. I would like to especially thank my mother Maryam, my late father, Shahram, and my brother Sohrab for all their sincere and unconditional support, love, and encouragement in all stages of my life. My accomplishments in my personal and academic life were unlikely without the support, love, and encouragement from my late grandfather Yusof, and my grandmothers Ezzat and Shahin. My extended family, especially my aunt Mahboubeh and her husband Ehsan, and my uncles Mehrdad, Hossein, and Alireza, have been a great source of support and encouragement during our most challenging times.

Lastly, I would like to thank my dearest and beloved wife Razieh for her endless love, support, and cheering. I would also like to thank my wife's family for their kindness and faith.

Date
Saeed Mehrang

SAEED MEHRANG

Saeed Mehrang holds a bachelor's degree in biomedical engineering from Azad University of Khomeinishahr as well as a master's degree in information technology from the Tampere University of Technology. His research interests include the analysis and applications of artificial intelligence for signal and image processing.

Table of Contents

Notations

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\boldsymbol{I}$	Identity matrix with dimensionality implied by the context

Indexing

a_i	Element i of vector $\boldsymbol{a}$
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$

Sets

$\mathbb{R}$	The set of real numbers
$\mathcal{X}$	The domain (or the sample space) of an arbitrary random variable

Functions

L	The training loss function
err	An arbitrary error function
Ω	The regularization function
$\mathcal{L}$	The likelihood function
log	The logarithm function with base 10
Pr	The probability of a random variable
$\mathbf{1}(c)$	The indicator function which returns 1 when the condition c is satisfied, and 0 otherwise
$\|\boldsymbol{x}\|$	L^2 norm of vector $\boldsymbol{x}$
$\mathfrak{X}$	DFT of signal (or vector) $\boldsymbol{x}$
$\langle \boldsymbol{a}, \boldsymbol{b} \rangle$	The dot product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$
$\boldsymbol{a} \cdot \boldsymbol{b}$	The dot product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$
$\boldsymbol{a} \odot \boldsymbol{b}$	Hadamard product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$

Datasets and Distributions

$\mathbb{X}$	A set of training examples (or the training set)
$\boldsymbol{x}^{(i)}$	The input vector of i^{th} observation from the training set
$y^{(i)}$	The output label of i^{th} observation from the training set

Abbreviations

AdaBoost	Adaptive boosting
AFib	Atrial fibrillation
ANN	Artificial neural network
BNP	B-type natriuretic peptide
CVDs	Cardiovascular diseases
CAD	Coronary artery disease
CART	Classification and regression tree
DSP	Digital signal processing
DFT	Discrete Fourier transform
ECG	Electrocardiogram
ESC	European Society of Cardiology
EU	European Union
GCG	Gyrocardiogram
HF	Heart failure
HFmrEF	Heart failure with mildly reduced ejection fraction
HFrEF	Heart failure with reduced ejection fraction
HFpEF	Heart failure with preserved ejection fraction
IMF	Intrinsic mode function
LTI	Linear time invariant
LSTM	Long short-term memory
MLP	Multilayer perceptron
NPs	Natriuretic peptides
NT-proBNP	N-terminal pro-B type natriuretic peptide
PPG	Photoplethysmogram
PMF	Probability mass function
RNN	Recurrent neural network
RobustBoost	Robust boosting
SCG	Seismocardiogram
SAMME	Stagewise additive modeling using a multi-class exponential loss function
SGD	Stochastic gradient descent
SVM	Support vector machines
XGBoost	Extreme gradient boosting

List of Original Publications

This dissertation is based on the following original publications, which are referred to in the text by their Roman numerals:

I Mojtaba Jafari Tadi, Saeed Mehrang, Matti Kaisti, Olli Lahdenoja, Tero Hurnanen, Jussi Jaakkola, Samuli Jaakkola, Tuija Vasankari, Tuomas Kiviniemi, Juhani Airaksinen, Timo Knuutila, Eero Lehtonen, Tero Koivisto, Mikko Pänkäälä. Comprehensive analysis of cardiogenic vibrations for automated detection of atrial fibrillation using smartphone mechanocardiograms. IEEE Sensors, 2018; 19(6):2230-42. (©2018 IEEE)

II Saeed Mehrang, Mojtaba Jafari Tadi, Tero Hurnanen, Timo Knuutila, Olli Lahdenoja, Jussi Jaakkola, Samuli Jaakkola, Tuija Vasankari, Tuomas Kiviniemi, Juhani Airaksinen, Tero Koivisto, Mikko Pänkäälä. Reliability of self-applied smartphone mechanocardiography for atrial fibrillation detection. IEEE Access. 2019; 7:146801-12. (©2019 IEEE)

III Saeed Mehrang, Olli Lahdenoja, Matti Kaisti, Mojtaba Jafari Tadi, Tero Hurnanen, Antti Airola, Timo Knuutila, Jussi Jaakkola, Samuli Jaakkola, Tuija Vasankari, Tuomas Kiviniemi, Juhani Airaksinen, Tero Koivisto, Mikko Pänkäälä. Classification of Atrial Fibrillation and Acute Decompensated Heart Failure Using Smartphone Mechanocardiography: A Multilabel Learning Approach. IEEE Sensors, 2020; 20(14):7957-68. (©2020 IEEE)

IV Saeed Mehrang, Mojtaba Jafari Tadi, Timo Knuutila, Jussi Jaakkola, Samuli Jaakola, Tuomas Kiviniemi, Tuija Vasankari, Juhani Airaksinen, Tero Koivisto, Mikko Pänkäälä. End-to-end sensor fusion and classification of atrial fibrillation using deep neural networks and smartphone mechanocardiography. Physiological Measurement, 2022; 43, 5: 055004. (©2022 IOPscience)

1 Introduction

Cardiovascular diseases (CVDs) cause the majority of deaths worldwide annually [1]. As defined by WHO, cardiovascular diseases are disorders that affect the heart and blood vessels. Nearly 18 million, or 32 percent of all worldwide deaths in 2019, were attributable to CVDs [1]. The main causes of CVD-related death were heart attacks and strokes, which contributed 85 percent of all CVD-related deaths in 2019 [1]. A total of 6 and 11 million people are diagnosed with cardiovascular diseases in the European Union (EU) and in Europe each year [2]. In 2015, the financial burden of the CVDs on the EU economy was estimated as high as 210 billion euros a year [3]. Among all the CVDs, atrial fibrillation (AFib) has caused substantial morbidity and mortality worldwide [4]. Heart failure, stroke, cognitive decline, depression, decline in quality of life, and hospitalization are among the potential outcomes of AFib [4]. As CVDs in general, and AFib in particular, are associated with significant health and financial burdens, it is imperative that we take immediate and careful action to prevent, diagnose, and treat these diseases as early as possible.

Despite the substantial morbidity and mortality rates, CVDs can be prevented by maintaining a healthy diet and a physically active lifestyle and by reducing smoking, tobacco use, and alcohol consumption [1]. Several studies have shown that preventive medicine can significantly reduce mortality rates, morbidity rates, health care costs, and hospital admissions. The outcomes of clinical intervention can also be improved by preventive medicine [5]. Prevention and management of CVDs can also benefit from technology. In recent years, remote monitoring of health and wellness through wearable devices has been adopted in a variety of studies and has been found to be effective for the prevention and treatment of CVDs [6; 7]. Utilizing these devices, we can continuously monitor the physical activity, heart rhythm, and blood pressure of the users. By analyzing these data, we can gain an understanding of the health of the cardiovascular system [8]. Furthermore, it can be used for tracking the efficacy of interventions and treatment plans as well as for improving future treatment plans. Remote cardiac monitoring, provided by digital health technologies, improves health outcomes while reducing hospital visits, transportation costs, and psychological burdens associated with diseases [9].

Adopting digital and remote health monitoring can lead to substantial benefits in the case of elderly care [10]. It is known that there is a direct correlation between the prevalence of CVDs and age [11]. CVDs are prevalent in approximately 70-

75% of 65 years or older individuals [11]. With the increasing lifespan of humans, the proportion of individuals over the age of 65 is growing. With such a growing population of elderly adults, the demand for healthcare and clinical services for the treatment of CVDs will grow even further which in turn increases healthcare costs and burdens [11]. With the adoption of digital and mobile health [12], wearable IoT devices [10], advanced analytical algorithms [13], and telemedicine [14], health care systems can not only control costs but also deliver better and more frequent preventive care that is backed by the insights extracted from the remotely collected data.

Wearable Electrocardiography and ambulatory photoplethysmography are probably the most popular sensing modalities that have been tested and shown valuable for CVDs and especially AFib detection [15]. Other alternatives for ambulatory CVD monitoring are seismocardiography and gyrocardiography which have been successfully used to monitor and classify a group of CVDs [16]. The mechanical movements or vibrations of the chest induced by the heart muscle can be recorded by triaxial accelerometer and gyroscope sensors [17]. The obtained signals are called seismocardiogram (SCG) and gyrocardiogram (GCG) which carry rich information about the heart functioning and in particular heart rate and heart rate variability, cardiovascular hemodynamics, and some cardiac diseases [17; 18]. The recording of SCG and GCG can be as simple as placing a smartphone on the chest while lying down and letting the built-in accelerometer and gyroscope sensors of the smartphone capture the chest movements [19]. The ubiquity of wearable sensors and smartphones in recent years has made it possible to record SCG and GCG signals out of laboratory settings and let the scientific community start investigating their utility in more detail [17].

As remote health monitoring and remote sensing technologies become more widespread, advanced data analysis techniques are becoming increasingly essential [20]. With the integration of signal processing and machine learning, powerful tools have been developed to meet this demand [21]. Using these tools, we have been able to successfully analyze SCG and GCG signals [17; 18]. With the increase in dataset size in the recent past, deep learning, which is a sub-field of machine learning, has gained increasing popularity and applicability to CVD-related data analysis [22], especially the detection of AFib [23]. In addition, as automated feature learning is possible through deep learning, engineers are able to spend less effort on feature engineering, which requires a lot of domain knowledge.

Since SCG and GCG signals can be recorded by smartphones outside of hospital settings, they may be suitable for remote monitoring of AFib, particularly when an electrocardiogram (ECG) recording cannot be acquired. Although SCG and GCG provide us with entirely different pieces of information compared with ECG, the provided data can be sufficient for the analysis and monitoring of some CVDs. Another significant CVD that requires special attention from the scientific community and healthcare systems is acute decompensated heart failure (ADHF) [24]. Undiagnosed

AFib can be one of the precursors of ADHF [24]. Tracking and monitoring ADHF outside clinical settings can substantially improve the efficacy of the treatment plans.

1.1 Objectives and Problem Statement

Throughout the course of this study, the main objective was to investigate the feasibility of using signal processing and machine learning techniques for the analysis of SCG and GCG signals primarily for the purpose of AFib detection. We were also interested in the investigation of the concurrent detection of AFib and ADHF utilizing SCG and GCG signals. The studied SCG and GCG signals were all gathered by smartphones and mostly from elderly patients who were admitted to the Department of Cardiology in Turku University Hospital [19; 25]. The results of this work can form a basis for designing powerful algorithms for the in-time detection and monitoring of AFib outside hospital settings utilizing ubiquitous sensing modalities such as smartphones. Accordingly, the objectives of this study were:

- **Objective 1**: classification of AFib using SCG and GCG signals via feature engineering and supervised learning.
- **Objective 2**: assessing the reliability of self-measured (or patient-applied) recording of SCG and GCG signals for the detection of AFib. Here, the term self-measured describes a recording that is fully carried out by the patients (or users) themselves.
- **Objective 3**: investigating the feasibility of concurrent detection of AFib and ADHF using a short SCG and GCG recording.
- **Objective 4**: classification of AFib using SCG and GCG signals via deep neural networks. Assessing the feasibility of automated end-to-end feature learning and classification.

Accordingly, the objectives of the research articles and their association with the thesis objectives are shown in Table 1.

Table 1. Association of the objectives and the research articles.

Publication	Obj. 1	Obj. 2	Obj. 3	Obj. 4
I	✓			
II	✓	✓		
III	✓		✓	
IV				✓

1.2 Thesis Overview

The thesis is organized as follows. Chapter 2 describes the cardiac physiology and cardiovascular diseases studied in this thesis work. Moreover, Chapter 2 describes the properties of a healthy heart (Section 2.1) as well as the properties of AFib and HF (Section 2.2). Chapter 3 briefly introduces the SCG and GCG signals. In the same chapter, the most popular family of pre-processing and feature engineering techniques are also presented (Sections 3.2 and 3.3). Chapter 4 introduces machine learning-based classification techniques. First, logistic regression (Section 4.1) and support vector machine (Section 4.2) are described. Next, tree-based classifiers (Section 4.3) including decision tree (Subsection 4.3.1), random forest (Subsection 4.3.2), adaptive boosting (Subsection 4.3.3), robust boosting (Subsection 4.3.4), gradient boosted trees (Subsection 4.3.5), and extreme gradient boosting (Subsection 4.3.6) are described. Finally, artificial neural networks (Section 4.4), convolutional neural networks (Subsection 4.4.1), and recurrent neural networks (Subsection 4.4.2) are described. Next, in Chapter 5, a brief overview of the original publications is provided. Lastly, in Chapter 6, the overall discussion and conclusions of this research study are presented.

1.3 Contributions

This thesis addresses the aforementioned research objectives and makes the following contributions:

- in-depth analysis of the use of feature engineering and supervised classification of multidimensional SCG and GCG signals for AFib classification (publications I, II, and III)
- designing pipelines for the analysis of biosignals using conventional supervised machine learning classifiers (publications I, II, and III)
- presenting a modern deep learning pipeline providing an end-to-end automated sensor fusion, feature learning, and classification for AFib detection (publication IV)

With the research studies of this thesis work, we showed how machine learning can help with the analysis of SCG and GCG signals and in particular for the detection of AFib. The results of this thesis work enabled a deeper understanding of SCG and GCG analysis through the lens of machine learning. It is worth mentioning that the presented techniques are general in the sense that they can be applied to similar biosignals and research studies.

2 Cardiac Physiology and Cardiovascular Diseases

2.1 Properties of a Healthy Heart

This section is widely based on the references [26; 27]. For better readability, repetitive referencing of these two sources is avoided in the rest of this section.

The heart is responsible for pumping blood to the body via an orchestrated contraction–relaxation cycle (or a cardiac cycle). The heart muscle (or myocardium) is composed of cardiomyocytes which are particular muscle cells designed for generating contractile force. In a healthy heart, the pumping procedure is regularly repeated many times per minute, each time causing one cardiac cycle. There are four hollow chambers in a healthy heart, two at the top and two at the bottom, which are referred to as atria and ventricles, respectively (see Fig. 1). The atria are smaller chambers that are responsible for receiving blood, while ventricles are responsible for sending blood to the organs of the body.

As can be seen in Fig. 1, a cardiac cycle consists of two major physiological phases, diastole, and systole. Diastole represents the phase during which blood is flowing toward the heart and is collected by the atria. In this phase, the ventricles are relaxed and blood is passively flowing from the left atrium into the left ventricle and from the right atrium into the right ventricle, respectively. During diastole, the right atrium receives oxygen-free (or venous) blood from the body through the superior vena cava and inferior vena cava. Similarly, the left atrium receives oxygenated (or arterial) blood from the lungs through four pulmonary veins. At the end of diastole, the atria contract and push the blood into the ventricles through the atrioventricular valves. Two valves are responsible for controlling the blood flow from the atria to the ventricles. These are called Mitral and Tricuspid, which are located between the left and right chambers of the heart, respectively. Systole represents the phase when the left and right ventricles contract and eject blood into the aorta and pulmonary artery, respectively. In this phase, the oxygenated blood is delivered to all the organs of the body, while the oxygen-free blood is sent to the lungs for refinement.

A healthy normal cardiac cycle is initiated by an electrical impulse that is fired by the sinoatrial node and causes the atria to contract. The passive blood flow from the atria to the ventricles is boosted by the extra push which is produced by atrial contraction. On ECG, an atrial contraction is seen as a P-wave, which is defined as a

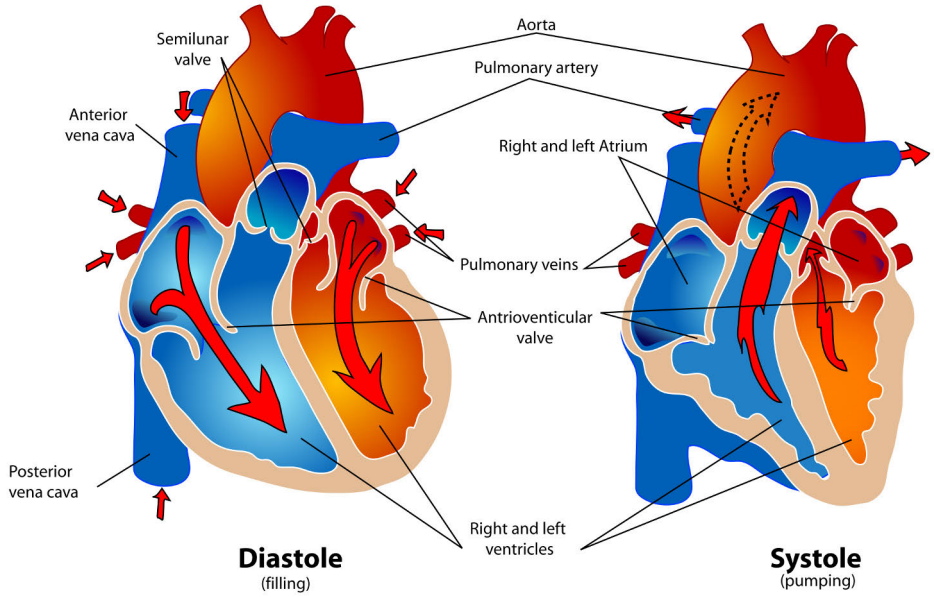


Figure 1. Anatomy of a healthy heart together with the demonstration of diastole and systole. *Animation created by Mariana Ruiz Villarreal, downloadable From Wikimedia Commons, the free media repository.*

response to depolarization of the atrial cardiomyocytes. When the ventricles are filled with blood, another electrical impulse, which is originated from the atrioventricular node, is fired. This causes the ventricles to contract and send the blood away from the heart through the aorta and pulmonary arteries. The ventricular contraction is caused by depolarization of the ventricular cardiomyocytes, which can be seen on the ECG as the QRS complex. As blood pressure within the ventricles increases due to the contraction of the ventricles, the atrioventricular valves close. Then, the ventricles eject the blood away from the heart through the aortic and pulmonic (or semilunar) valves. The blood starts to flow through the body. The blood ejection continues until the end of ventricular cardiomyocytes repolarization. This is the point when the blood pressure inside the ventricles drops below the aortic pressure. As a result, the semilunar valves get closed, and soon after, the atrioventricular valves get opened again. This is the beginning of the next cardiac cycle. The ventricular repolarization can be seen on ECG with a waveform known as T-wave.

2.2 Cardiovascular Diseases and Disorders

2.2.1 Atrial Fibrillation

AFib is the most common sustained heart arrhythmia in adults worldwide [4] and is associated with considerable mortality and morbidity [4]. The prevalence of AFib is approximately 1.0% in the general population and increases to 8% among people over the age of 80. Hypertensive heart disease, metabolic syndrome, mitral valve disease, and coronary artery disease are among the underlying causes of AFib [28]. Death, stroke, HF, cognitive decline, depression, declined quality of life, and hospitalization are among the potential AFib-related outcomes that altogether impose a substantial burden on patients and health systems [4]. There are more than 33 million individuals worldwide who suffer from AFib [29]. The incidence and prevalence of AFib have increased with an increase in its associated mortality. The incremental national cost of AFib has been estimated at 26 billion dollars in the United States alone [29].

Persistent AFib puts patients at risk of systemic embolism. As a result of fibrillating atria, blood clots form easily [28]. A stroke or a vascular occlusion can occur if pieces of these clots embolize and travel through the systemic circulation. The risk of stroke for those with non-rheumatic AFib is high, especially if they have congestive HF, hypertension, diabetes mellitus, or a history of transient ischemic attacks or strokes [28].

AFib is defined as a supraventricular tachyarrhythmia caused by randomly initiated electrical impulses that cause fibrillation of the atria rather than a proper contraction [4]. When a healthy heart is functioning normally, a single electrical impulse initiated by the SA node coordinates atrial and ventricular contractions. The electrical impulse is transmitted through the heart's electrical conduction system, composed of the atrioventricular node, bundles of His, bundle branches, and Purkinje fibers. Electrical impulses in AFib do not originate from the SA node but are instead generated by ectopic sites located in and around the atria. The atria do not contract properly but instead fibrillate. There are many uncoordinated ectopic electrical impulses bombarding the atrioventricular node, which only allows a fraction of these impulses to pass through and reach the ventricles. The atrioventricular node is unable to maintain the regular impulse generation, leading to uncoordinated and irregularly irregular contractions of the ventricles [4]. One of the key characteristics of AFib is the irregularity of ventricular contractions, which can be observed in a wide variety of biosignals [25].

Clinical AFib diagnosis is done via ECG and it requires a minimum of 30 seconds of AFib on at least one ECG lead. On ECG, AFib is characterized by the absence of P-wave and the presence of irregularly irregular R-R intervals [4]. On SCG and GCG signals, AFib can be seen with irregular and random motion patterns, which can be attributed to irregular ventricular contractions and aortic openings [25]. AFib diag-

nosis is, however, a difficult task if AFib is not the dominant heart rhythm. In detail, there are two main variants of AFib, persistent (or symptomatic) and paroxysmal (or asymptomatic). In the persistent case, AFib is present predominantly and manifests itself with palpitations, chest tightness or pain, poor effort tolerance, dizziness, syncope, and sleep disorder [4]. However, a significant minority of sufferers experience none of these symptoms [28]. In the paroxysmal case, AFib is only present for a short period, leaving the sufferers unaware because of stable hemodynamics. Detection of paroxysmal AFib is hardly possible through short infrequent checkups in clinical settings. With the advent of long-term remote cardiac monitoring, the detection of paroxysmal AFib has become feasible as users can frequently and even continuously monitor the cardiac operations and in particular the cardiac rhythm almost anywhere [4].

2.2.2 Heart Failure

HF is a clinical cardiovascular syndrome that manifests itself with symptoms such as breathlessness, ankle swelling, and fatigue [24]. In plain terms, HF can be regarded as a condition where the heart muscle partially loses its power to pump blood in either systolic, diastolic, or both [24]. There is a wide spectrum of conditions that can cause HF including coronary artery disease (CAD) and hypertension as the two most prevalent factors in developed countries as well as valve diseases, arrhythmias, cardiomyopathy, congenital heart disease, endomyocardial disease, pericardial disease, metabolic and neuromuscular diseases [24]. The symptoms and signs of HF may be very similar to those of non-CVDs, such as anemia, pulmonary, renal, thyroid, or hepatic diseases. These signs and symptoms are only indicative of HF in the presence of cardiac dysfunction [24]. According to estimates, 26 million people worldwide were diagnosed with HF in 2012, costing 108 billion dollars annually [29]. By 2030, the total cost of direct medical treatment for HF in the United States will rise from 21 billion dollars to 53 billion dollars [29].

The European Society of Cardiology (ESC) [24] indicates that HF can be divided into three subgroups based on the left ventricular ejection fraction (LVEF). The HF with reduced ejection fraction (HFrEF) category is designated in the case of a significant reduction in left ventricular systolic function, e.g. LVEF less than or equal to 40%. When there is a mild reduction in left ventricular systolic function, e.g. LVEF between 41% and 49%, HF with mildly reduced ejection fraction (HFmrEF) is designated. In addition, when the LVEF value is greater than or equal to 50% and there is evidence of structural and/or functional cardiac abnormalities and/or elevated natriuretic peptides (NPs), HF with preserved ejection fraction (HFpEF) is designated.

Chronic HF can also be classified into compensated and decompensated [30]. An HF that is compensated appears stable, with little evidence of fluid retention or pulmonary edema [30]. A decompensated HF is characterized by breathlessness on

exertion, either in the acute phase or in the chronic phase [30]. Decompensated HF may manifest as pulmonary oedema or lethargy and malaise. Decompensation may be caused by recurrent ischaemia, arrhythmias, infections, and electrolyte imbalances [30]. In the case of acute decompensated HF (ADHF), the symptoms of HF are severe-enough to require unplanned hospitalization, emergency room visits, or office visits [31]. ADHF is characterized by pulmonary and systemic congestion due to increased left- and right-heart filling pressures [31]. The majority of ADHF hospitalizations occur as a result of worsening chronic HF. However, approximately 15% to 20% of ADHF admissions are a result of new diagnoses of HF [31]. The average age of hospitalized patients in the U.S. is 70 to 75, with both sexes equally represented. Approximately half of the hospitalized patients with HF have an LVEF less than 0.4, which indicates that their heart has moderate to severe difficulty filling their left ventricle [31]. HF patients who are newly diagnosed are much more likely to have pulmonary oedema or cardiogenic shock, while chronic HF patients who are decompensating typically have weight gain, exertional dyspnea, and orthopnea [31]. In addition to coexisting valvular diseases and dilated cardiomyopathy, AFib or atrial flutter is remarkably common in about 30%–46% Of patients with ADHF [31].

HF is diagnosed using several measurements, including an electrocardiogram, echocardiography, and clinical examination, as well as laboratory testing for the N-terminal pro-B type natriuretic peptide (NT-proBNP) and B-type natriuretic peptide (BNP). According to the ESC, the recommended diagnostic procedure is the clinical examination, ECG inspection, and NT-proBNP and BNP tests. If there are any abnormalities suggesting HF, echocardiography may be required. When there is evidence of abnormality in all the examinations carried out, HF is confirmed and then the phenotype is determined based on LVEF [24].

A number of studies from several countries have shown that survival rates for HF patients improved dramatically between 1980 and 2000, but this trend may have plateaued since then [24]. Despite the improvements in the prognosis of HFrEF since the first treatment trials decades ago, the quality of life and the overall prognosis of patients remain poor [24]. These statistics indicate that it is essential to identify risk factors and early signs of HF better and more rapidly. Furthermore, continuous monitoring of the efficacy of the treatment interventions and the prognosis is critical and may be enhanced by the use of modern remote monitoring technologies [24; 32].

2.2.3 Comorbid Atrial Fibrillation and Heart Failure

AFib and HF can reciprocally influence each other through a range of mechanisms, establishing a cycle of interdependence and exacerbation [33]. HF can initiate AFib by causing increased filling pressures, diastolic dysfunction, mitral regurgitation, and the activation of neurohormonal pathways. These factors can cause structural and electrical remodeling of the atria [34]. Conversely, AFib can predispose individu-

als to HF due to the detrimental effects of rapid and irregular heart rate, diminished atrial contraction, compromised hemodynamic performance, and neurohormonal activation [35]. Additionally, AFib and HF share common risk factors and pathogenetic mechanisms, including age-related changes, cardiometabolic abnormalities, and systemic inflammation, which concurrently contribute to their development [36].

Patients with existing HF are likely to experience a doubled risk of death when they develop AFib, while those with existing AFib will experience a tripled risk of mortality when they develop HF [29]. In spite of intensive investigation, the pathophysiological interactions between AFib and HF remain incompletely understood. Furthermore, HFpEF has emerged as a significant disease entity in recent years, and there is limited knowledge about its relationship to AFib, which differs from that of HFrEF [29]. It is currently unknown what is the optimal treatment strategy for patients with both AFib and HF, and guidelines are continually evolving [29].

The management of comorbid AFib and HF aims to achieve various key objectives, including the restoration and maintenance of sinus rhythm, effective control of ventricular rate, prevention of thromboembolic events, and optimization of HF therapy [37; 38]. The selection of appropriate treatment strategies depends on individual patient characteristics and response to medications, encompassing options such as pharmacological or electrical cardioversion, administration of antiarrhythmic drugs, employment of rate-control medications, utilization of anticoagulants, consideration of catheter ablation procedures, or even surgical interventions [37; 38]. Nonetheless, it should be noted that the current body of evidence from randomized trials is relatively limited in guiding the optimal management approach for AFib in patients with coexisting HF, and some of the available therapeutic interventions may entail potential adverse effects or demonstrate limited efficacy within this specific patient population [37; 38]. It is worth noting that there are several unmet clinical needs for the management of these two conditions, such as improving the accuracy and timeliness of diagnosis, optimizing the pharmacological and device-based therapies, enhancing patient adherence and self-care, and reducing hospitalization and mortality rates [39]. Therefore, the management of AFib in patients with HF requires a comprehensive approach that considers the type, duration, and severity of AFib, the underlying cause, and stage of HF, the patient's symptoms and preferences, and the potential benefits and risks of different therapeutic options [39].

Remote sensing has emerged as a promising technology to address these needs. Remote sensing can enable continuous and personalized monitoring of cardiac rhythm, blood pressure, oxygen saturation, activity level, and other relevant parameters for patients with AFib and HF. Remote sensing can also facilitate the timely detection of arrhythmias, exacerbations, medication adherence, and response to therapy. Despite the potential benefits, there is not enough clinical evidence that comprehensively proves the applicability of remote sensing technologies for a complex disease such as HF [40]. Issues related to device accuracy, clinical validity, the ab-

sence of standardized regulatory policies, concerns regarding patient privacy, and the optimal use of the devices are impeding the widespread adoption of remote sensing and wearable technologies [40].

3 Seismocardiogram and Gyrocardiogram Signals

This chapter briefly describes what cardio-physiological variables we can extract and analyze by SCG and GCG signals. Moreover, we briefly describe how SCG and GCG signals are collected, pre-processed, and analyzed. In the next three sections, first, the general idea behind the collection of SCG and GCG signals and their applications are provided. Next, a brief overview of noise removal and signal enhancement techniques is presented. Ultimately, a few of the fundamental features that are usually implemented for this type of data are concisely described.

3.1 Modeling Mechanical Functioning of the Heart

The electrophysiological activity of the heart as measured by ECG is the most popular and possibly the richest source of information for analyzing the functioning of the heart. The timing, shape, and amplitude of the P, Q, R, S, and T waves all carry vital information about the underlying fitness, potential abnormalities, or disorders of the heart. Besides ECG, the heart can be monitored mechanically through SCG and GCG. Similarly, we can monitor the operations of the heart audibly through a phonocardiogram. Samples of all these signals together with the annotation of waveforms and fiducial points are plotted in Figure 2. It is worth mentioning that the joint measurement of SCG and GCG is also called mechanocardiography in some of our research articles.

The contractions and relaxations of the myocardium cause movements on the chest wall which are recordable by motion capture sensors. These movements have been shown to have correlations with some of the underlying mechanical functions of the heart (see Fig. 2) [41]. By placing an accelerometer on the chest, one can capture the seismic vibrations of the chest wall [42]. The acquired signal is SCG [42]. By placing a three-dimensional accelerometer sensor on the chest, the chest vibrations can be captured in all three spatial dimensions [41]. The captured vibrations in the anteroposterior axis of the body are, though, the most known due to the clearer correspondence of the signal waveforms and the underlying physiology of the heart (see SCG signal in Fig. 2). In principle, SCG captures the acceleration of the chest wall; however, one can capture the angular velocity of the chest wall as well. For this purpose, a gyroscope sensor can be used. Similar to SCG, the acquired signal

is GCG and has been shown to correlate with the underlying mechanical functions of the heart [43]. Scientists have shown that in a healthy normal heart, there are particular waveforms and fiducial points in SCG and GCG signals that are correlated with mitral and aortic valve openings and closures, pre-ejection period, isovolumetric contraction time, isovolumetric relaxation time, left ventricular ejection time, electromechanical delay, total systolic time, Q-wave to MO duration [44; 41; 43]. Some of these physiological variables are marked on SCG and GCG signals in Fig. 2.

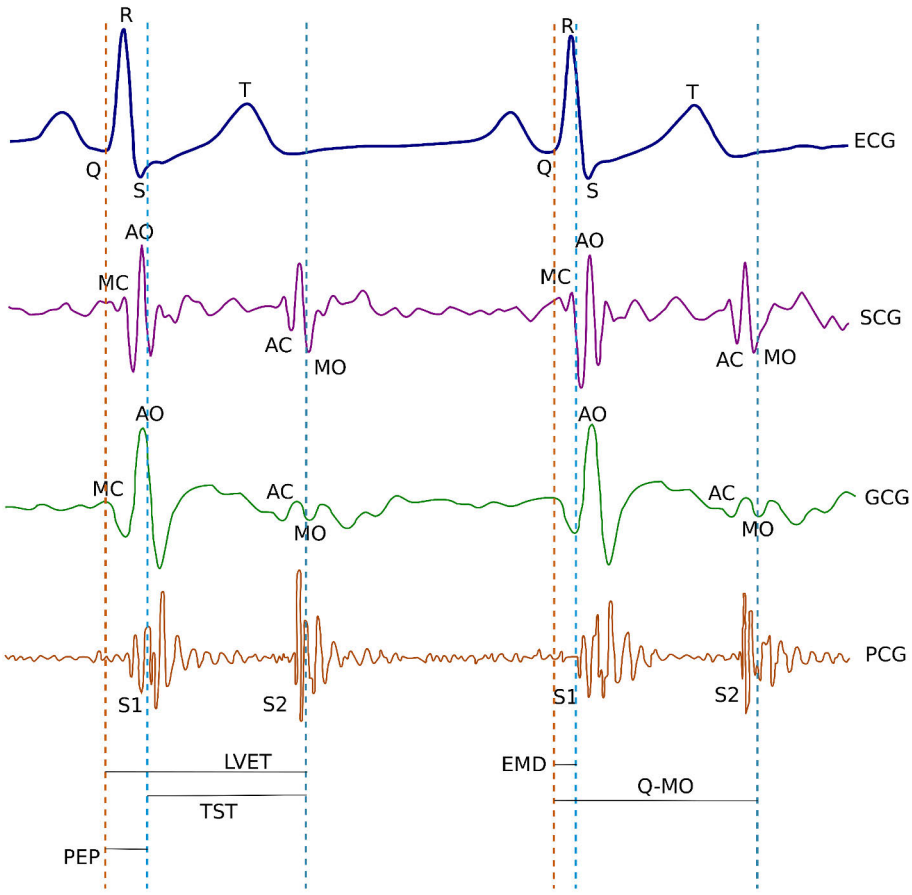


Figure 2. Concurrent demonstration of ECG, Z-axis of SCG, Y-axis of GCG, and phonocardiogram (PCG) signals together with some of the fiducial points representing cardio-physiological functioning of a healthy heart. The annotated conditions are mitral and aortic valve openings (MO and AO) and closures (MC and AC), pre-ejection period (PEP), isovolumetric contraction time (IVCT), isovolumetric relaxation time (IVRT), left ventricular ejection time (LVET), electromechanical delay (EMD), total systolic time (TST), Q-wave to MO duration (Q-MO). Image reproduced from [18] which is an edited version of the signals originally published in [43; 45]. The cited works are available under the license CC-BY 4.0.

SCG and GCG signals can be recorded noninvasively and unobtrusively by the

widespread MEMS motion sensors [16]. Nowadays, MEMS motion sensors are widely used inside smart consumer electronic devices such as smartphones and smartwatches for non-medical applications. However, one can utilize these devices for recording SCG and GCG signals. A straightforward scenario can be placing a smartphone on the chest of an individual who is lying in the supine position and starting logging the tri-axial accelerometer and gyroscope sensor values [19]. Such a simple scenario has been used for collecting SCG and GCG signals for the classification of AFib [19]. In this thesis work, the same smartphone-derived SCG and GCG signals have been utilized as the data source for investigating different research questions covering machine learning-based AFib and ADHF classification.

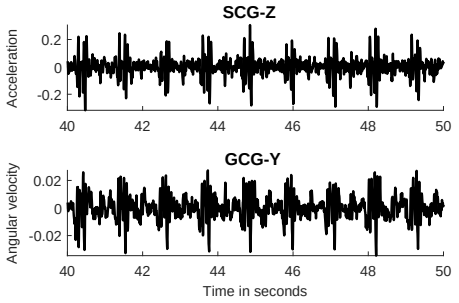
Figure 3 displays representative samples of smartphone SCG and GCG cardiac waveforms. The waveforms depicted in the figure correspond to three distinct disease groups within the analyzed dataset. These groups include (a) controls, representing individuals without AFib or ADHF; (b) AFib cases without ADHF; and (c) cases presenting both AFib and ADHF. Upon examination, it is evident that the control waveform exhibits a regular rhythm with monomorphic repeating patterns in both the rotational and translational signals. In contrast, the waveforms from the [AFib, non-ADHF] and [AFib, ADHF] conditions demonstrate irregular rhythms and abnormal morphological characteristics.

All types of biosignals often contain noise which ideally needs to be removed and separated from the physiological data generation process [47; 48; 49]. Often times the magnitude of the noise is too large that the majority of the critical information becomes hardly accessible. Consequently, there have been decades of research and development work behind the inventions of noise removal techniques which were solely designed and investigated for biosignal processing [49; 48]. Likewise, SCG and GCG signals contain noise and need to be enhanced prior to any information extraction or any further analysis [16]. In the next section, this topic is briefly introduced.

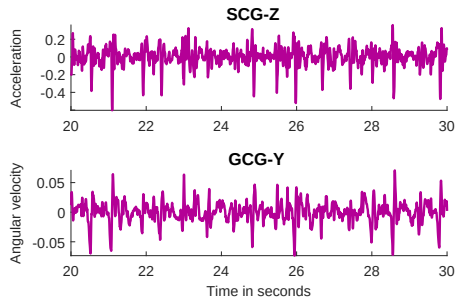
3.2 Noise Removal and Signal Enhancement

A variety of noise sources, including motion artifacts, environmental vibrations, and sensor mechano-electronics, usually affect SCG and GCG signals [17]. Motion artifacts are the most problematic and prevalent source of noise. During the recording, any movement of the user's body, sensors, or the environment where the user is located can cause motion artefacts [17]. These artifacts can sometimes be stronger than the actual vibrations induced by the heart on the chest wall [17]. Therefore, noise removal is a crucial step in the analysis of SCG and GCG signals. It is worth noting that depending on the magnitude of the noise, some measurements might become totally useless even if advanced noise removal techniques are applied.

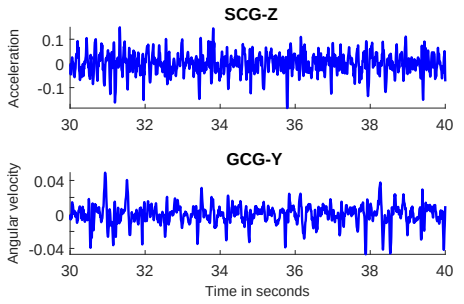
Conventional band-pass filtering is the first and the most used option for remov-



(a) Control, i.e. no AFib or ADHF.



(b) AFib without ADHF.



(c) AFib with ADHF.

Figure 3. Example signals of the subjects in the three possible categories namely (a) Control, i.e. no AFib or ADHF, (b) AFib without ADHF, as well as (c) AFib with ADHF. Images adopted from [46] (©2020 IEEE).

ing motion artifacts caused by unwanted movements such as breathing [17; 16]. Normalized least mean square adaptive filter was used in another study [50] to remove motion artifacts from moving subjects. More sophisticated signal decomposition techniques such as empirical mode decomposition [51] and ensemble empirical mode decomposition [52] were used to decompose the signal into different intrinsic mode functions (IMFs). These methods work by reconstructing the signal using IMFs which contain desired physiological information while leaving out the IMFs which mostly represent the noise. Another powerful noise removal by signal decomposition is singular spectrum analysis [25] to discard the noisy components and subsequently reconstruct the noise-free signal from the remaining components. When only the location of heartbeats in time is of interest, envelope extraction [25] can be a powerful technique to deliver a signal with spikes at each heartbeat. Once noise removal is done, we need to characterize the input signals in order to infer the heart's condition. This part is usually done by feature engineering which is briefly introduced in the next section.

3.3 Feature Engineering

For many decades scientists have been investigating biomedical signals by computing mathematical transformations that were designed to reveal characteristics of the target physiological phenomena [53]. The outputs of these mathematical transformations are the so-called features that are used as inputs for machine learning algorithms or statistical analysis systems. The process in which effective features are identified, computed, and investigated is called feature engineering and/or feature extraction [53]. Obviously, finding the most optimal features requires a lot of domain knowledge and experience of the problem at hand [16].

Depending on the type of the mathematical transformation, feature extraction techniques for signal data can be categorized into (1) time domain, (2) frequency domain, and (3) joint time-frequency domain [53]. There is a plethora of research studies that have investigated the utility of each of the above-mentioned categories for the characterization of SCG signals [41; 16].

In the time domain, statistical properties, statistical model fitting, fiducial points, heart rate variability, various measures of signal entropy, empirical mode decomposition, singular spectrum analysis, independent component analysis, and cardiac time intervals are among the most popular features and transformations applied to SCG signals [16]. In the frequency domain, the Fourier transform is usually computed, and from the resulting Fourier domain signal, statistical properties, power spectral density of different frequency bands, various measures of entropy, as well as the magnitude and phase properties are calculated [16]. It is worth mentioning that Hilbert transform has been also used in some of the prior studies [54]. In the time-frequency domain, short-time Fourier transform, as well as discrete and continuous wavelet transform, have been used to analyze the frequency content of an input signal in various time windows [41; 16].

Biosignals are usually non-stationary. In other words, their properties vary over time because of the variations in the underlying physiological functioning of the organ that is being monitored. In the field of cardiac monitoring, the heart function can vary from one second to another due to the effects of the sympathetic nervous system, the hemodynamic status, and disorders such as arrhythmia. In this case, it is crucial to be able to characterize the signals in such a way that we can extract and reveal the aforementioned non-stationary properties.

Heart rhythm contains a wealth of information concerning the heart's fitness and well-being. Arrhythmias, encompassing different types of rhythm disorders, include conditions like AFib that necessitate prompt medical attention. AFib causes irregularities in the heart rhythm which could be described by frequency and time-frequency analysis of the signals on which heartbeats are visible. In the next paragraphs, some of the fundamental features or transformations that are utilized to reveal the properties of the heart rhythm are briefly described. Please note that these

are only a small set of all the possible mathematical transformations that we can use to analyze heart rhythm and biosignals in general.

3.3.1 Histogram analysis

Histograms represent the approximation of the probability mass function (PMF) of discrete signals. Once we have an approximation of the PMF, we can measure its various features. For instance, we can measure various orders of moments including the expected value, variance, skewness, and kurtosis. The different entropy measures could also be computed for the obtained histogram. These features can then be used as inputs to machine learning models.

The creation of a histogram requires defining a finite number of non-overlapping bins over the range of the signal. Each bin refers to an interval over the range of the signal. With more bins, the approximation becomes more accurate. However, in the case of signals of short length, the number of bins should be selected carefully and empirically according to the properties of the signal. Once the bin number is defined, we count the number of signal values that fall into each interval. The histogram is usually sketched by adjacent bar graphs each representing the calculated counts.

If we set N to be the length of an arbitrary signal and K to be the total number of histogram bins, the histogram values m_i should satisfy Equation 1.

$$N = \sum_{i=1}^{K} m_i \tag{1}$$

3.3.2 Entropy

In physics, entropy quantifies a system's disorder, randomness, or uncertainty. In information theory, entropy quantifies the amount of uncertainty or surprise of a system as measured by the probabilities of each outcome of the system. For a scalar random variable x, the probabilities of the outcomes are usually modeled by a normalized histogram. In mathematical terms, entropy $H(\mathrm{x})$ is defined as follows,

$$H(\mathrm{x}) = -\sum_{x \in \mathcal{X}} p(x)\ log(p(x)) \tag{2}$$

where $\mathcal{X}$ holds the set of all possible values of the random variable x. The definition presented in Equation 2 is known as Shannon entropy [55].

When it comes to signal data, Shannon entropy is unable to distinguish between a regular and fully irregular signal whose elements are generated from the same Bernoulli PMF. As a result, approximate entropy [56] and sample entropy [57] were introduced.

As mentioned above, entropy is a measure of the randomness of the system or input signal. Hence, it can be used as a feature to characterize the properties of biosignals. For instance, using entropy we can measure the amount of randomness of (1) a short signal segment in the time domain, (2) the frequency spectrum of a signal, (3) the frequency spectrum of its sub-segments, or (4) each component of a decomposed signal.

3.3.3 Convolution

At the heart of digital signal processing (DSP), there is a mathematical operation called convolution based on which many of the systems are described [58]. It is through convolution that we can compute the output of linear time-invariant (LTI) systems for every given input provided that the impulse response of the LTI system is accessible [58]. In simple terms, convolution is the operation that allows us to combine the input signal $\boldsymbol{x}$ and the impulse response $\boldsymbol{h}$ to form a third signal $\boldsymbol{y}$ which is formally written as $\boldsymbol{y} = \boldsymbol{x} * \boldsymbol{h} = \boldsymbol{h} * \boldsymbol{x}$.

We can view convolution operation from the input signal perspective as a decomposition and synthesis operation. In detail, the input signal is first decomposed into shifted and scaled impulses, and then, given the impulse response of the LTI system, a shifted and scaled version of the impulse response is created corresponding to each input impluse [58]. At last, all the shifted and scaled impulse responses are synthesized (summed) to form the output signal [58].

In mathematical form, convolution operation can be viewed from the output signal perspective and written as follows,

$$y_i = \sum_{j=0}^{M-1} h_j \, x_{i-j} \ , \quad i \in \{0, ..., N + M - 2\} \tag{3}$$

for an M-point real-valued impulse response $\boldsymbol{h}$ and an N-point real-valued input signal $\boldsymbol{x}$. In this formulation, all the negative indices contain zero values. Similarly, all the indices outside the domain of each signal contain zero values as well. As can be seen in Equation 3, for every sample of the output signal, we need to compute a dot product of the two vectors of both lengths M. The first vector is the impulse response $\boldsymbol{h}$, and the second one is a shifted and time-reversed M-point segment from the input signal $\boldsymbol{x}$. This formulation is known as *the convolution sum*. To enable the convolution machine to work properly in practice, we must pad the input signal $\boldsymbol{x}$ with $M - 1$ zeros on both ends. The output signal will be of length $N + M - 1$.

While convolution, as such, is not directly used for feature extraction due to its computational complexity, it is a fundamental transformation in many feature extraction techniques.

3.3.4 Correlation in DSP

Similar to convolution, correlation in DSP is a mathematical operation that combines two signals to produce a third signal. This third signal is called cross-correlation if the two input signals are different. In the case of correlating a signal with itself, the resulting signal is referred to as autocorrelation [58]. With cross-correlation operation, we are usually after determining where (or if) a signal occurs in another signal [58].

For detecting known waveforms in random white noise, cross-correlation is the most effective technique. Cross-correlation produces a peak that is higher above the noise than any linear system when a similar pattern in the two signals occurs [58]. The process of using cross-correlation to detect a known waveform is commonly referred to as matched filtering [58]. Using autocorrelation, it is possible to identify periodic signals obscured by noise as well as the unrecognized fundamental frequency of a signal [58].

Cross-correlation can be expressed in a mathematical form that looks very similar to convolution, although it is a slightly different operation [58]. Similar to convolution, we pad the input signal with zeros on both ends. But, unlike convolution, we do not reverse the input signal. The cross-correlation operation can be written in the following mathematical form,

$$y_i = \sum_{j=0}^{M-1} h_j \; x_{i+j} \; , \quad i \in \{0, ..., N + M - 2\} \tag{4}$$

for an M-point real-valued signal $\boldsymbol{h}$ and an N-point real-valued signal $\boldsymbol{x}$. In this formulation, all the negative indices contain zero values. Similarly, all the indices outside the domain of each signal contain zero values as well. As can be seen in Equation 4, for every sample of the output signal, we need to compute a dot product of the two vectors of both lengths M. The first vector is the $\boldsymbol{h}$ and the second one is a shifted M-point segment from the signal $\boldsymbol{x}$.

3.3.5 Discrete Fourier Transform

The discrete Fourier transform (DFT) is one of the Fourier analysis techniques devised for discrete or digital signals [58]. Using DFT, one can decompose a digital signal into a finite set of sinusoidal waves. In detail, a discrete signal of length N can be decomposed into orthogonal $\frac{N}{2} + 1$ sine and $\frac{N}{2} + 1$ cosine waves each of which with a different frequency and amplitude. The frequencies of these waveforms run from zero all the way up to Nyquist frequency which is equal to half of the signal's sampling frequency [58]. The amplitudes of these waveforms represent how strong each frequency component is within the signal of interest. The signal of interest is

usually defined as a random variable that changes over time. Hence, the DFT of such a signal results in a transformation from the time domain to the frequency domain.

From an algebraic perspective, each of those sinusoidal waves is a basis function (vector) and with the DFT we want to project an input signal into the domain that is spanned by those basis functions. These basis functions are shown in Equation 5 below,

$$\begin{aligned} C_{k,i} &= cos(\frac{2\pi ki}{N}) \\ S_{k,i} &= sin(\frac{2\pi ki}{N}) \end{aligned} \tag{5}$$

where N is the length of the input signal, i is the time index running from zero to $N-1$, and k is the frequency index running from zero to $\frac{N}{2}$. Here, $C_{k,i}$ and $S_{k,i}$ represent the cosine and sine wave samples carrying frequency component k at time index i.

To decompose the input signal into these basis functions, we need to calculate the amplitude of the basis functions. This is done via measuring the dot product of the input signal and each of the basis functions as shown in Equation 6 below,

$$\begin{aligned} Re\mathfrak{X}_k &= \sum_{i=0}^{N-1} x_i \; cos(\frac{2\pi ki}{N}) \\ Im\mathfrak{X}_k &= -\sum_{i=0}^{N-1} x_i \; sin(\frac{2\pi ki}{N}) \end{aligned} \tag{6}$$

where $Re\mathfrak{X}$ and $Im\mathfrak{X}$ are the real part (or the amplitude of the cosine waves) and imaginary part (or the amplitude of the sine waves) of the DFT output, respectively. The index k of the real part of the DFT output denotes the amplitude of the cosine wave carrying frequency component k. Similarly, index k of the imaginary part of the DFT represents the amplitude of sine wave carrying frequency component k [58].

Once the amplitudes of all basis functions are calculated, we can synthesize the input signal $\boldsymbol{x}$ using Equation 7[1].

$$x_i = \sum_{k=0}^{\frac{N}{2}} Re\mathfrak{X}_k \; C_{k,i} + \sum_{k=0}^{\frac{N}{2}} Im\mathfrak{X}_k \; S_{k,i} \tag{7}$$

The illustration of the DFT output using $Re\mathfrak{X}$ and $Im\mathfrak{X}$ is called rectangular notation in the literature. It is worth knowing that there is an equivalently important

[1] This synthesis equation is not entirely correct. The $Re\mathfrak{X}$ and $Im\mathfrak{X}$ need to be scaled by scaling factors that are left out here. Please see [58] for more information.

alternative to rectangular notation which is called polar notation. From trigonometry, we know that $a\cos(x) + b\sin(x) = c\cos(x + \theta)$. In polar notation, the goal is to utilize this conversion to display the DFT synthesis represented in Equation 7 in terms of two new variables c (or magnitude) and θ (or phase angle). With polar notation, we can visualize and understand the DFT output more easily than the rectangular notation. To change from rectangular to polar notation we can follow the transformation presented in Equation 8 [58].

$$\begin{aligned} Mag\mathfrak{X}_k &= (Re\mathfrak{X}_k^2 + Im\mathfrak{X}_k^2)^{\frac{1}{2}} \\ Phase\mathfrak{X}_k &= \arctan\left(\frac{Im\mathfrak{X}_k}{Re\mathfrak{X}_k}\right) \end{aligned} \tag{8}$$

where $Mag\mathfrak{X}$ denotes the magnitude of the DFT and $Phase\mathfrak{X}$ denotes the phase angle [58].

Using DFT, we can create the frequency spectrum of a signal by turning the DFT output into polar notation and inspecting the magnitude of the DFT. The frequency components that are present in the input signal are usually pronounced with peaks extending above the amplitude of the noise [58].

Convolution is used to analyze systems in the time domain. Similar analyses can be performed in the frequency domain via DFT. We know that every input signal can be decomposed using DFT into cosine waves with a specific amplitude and phase. The same procedure can be done on the output signal of a system. By doing so, you can completely describe any linear system based on how cosine waves flow through it. Such a description of a system is called the system's frequency response [58]. A system's frequency response is determined by the Fourier Transform of its impulse response. The convolution operation in the time domain is a computationally intensive transformation. Convolution in the time domain is replaced with multiplication in the frequency domain [58] and likewise, deconvolution is replaced with simple division [58]. Such a property allows us to study LTI systems more easily in the frequency domain rather than the time domain, especially in favor of computational complexity. Thanks to a technique called Fast Fourier Transform (FFT) that implements DFT in a substantially efficient manner, the transition from the time domain to the frequency domain can save us both time and computation [58].

Many of the feature engineering techniques that are implemented for characterizing digital signals, and in particular biosignals, are derived from one or more of the core features or transformations that were described in this chapter. Interested readers are encouraged to see the authors' original publications reprinted in this thesis for more information about each feature type used for the analysis of SCG and GCG signals throughout the author's study.

4 Supervised Classification Algorithms

In supervised learning, a classification model describes the mathematical function that approximates a mapping between the label y and the input $\boldsymbol{x}$. As an example, we consider a linear parametric model, where the prediction $\hat{y}$ is computed as the dot product of model parameters $\boldsymbol{\theta}$ and input feature vector $\boldsymbol{x}$, i.e. $\hat{y} = \langle \boldsymbol{\theta}, \boldsymbol{x} \rangle$. Depending on the task and the classification model, the prediction value may have different interpretations. For instance, in a logistic regression classifier, the output is always the probability of the positive class. In an artificial neural network classifier, the output of the model can be an unnormalized activation of a perceptron. The parameters of the model $\boldsymbol{\theta}$ are learned through a training process which may require some form of iterative numerical optimization. For a parametric machine learning algorithm [59], the training process involves finding the values of $\boldsymbol{\theta}$ that best fit the training data. In this case, to train the model, we need to define a *loss function* $L(\boldsymbol{\theta})$, which measures how well the model fits the training data. The function L usually quantifies the training error or the goodness of fit. Optionally, the loss function may contain some additional terms for regularization – or model complexity penalty.

In contrast to parametric machine learning algorithms, a nonparametric machine learning algorithm does not rely on strong assumptions about the form of the mapping function [59]. By making no assumptions, the algorithm is generally able to learn any form of mapping function from the training data. The mapping function these algorithms provide is, however, dependent on the input data [59]. In other words, by changing the input data, the mapping function may completely change.

In the following sections, supervised classification techniques that were used throughout this thesis work are concisely described. The techniques are logistic regression, support vector machine, random forest, robust boosting, extreme gradient boosting, and artificial neural networks.

4.1 Logistic Regression

Logistic regression is a specific type of generalized linear model that is designed to model a dependent variable that takes a finite set of categorical values [60]. Logistic regression is primarily designed for the estimation of a mapping between a set of independent variables and a binary dependent variable – binary logistic regression – which can be used to perform binary classification. There exists a multinomial

logistic regression implementation as well that enables the prediction of a dependent variable that takes more than two possible discrete outcomes [60].

The binary logistic regression algorithm differs from ordinary least square regression in a way that instead of finding a direct mapping from the independent variables to the dependent variable, it tries to estimate a linear mapping to the logarithm of the odds (or log-odds) of the dependent variable [60]. In other words, the algorithm relies on regressing the log-odds of the dependent categorical variable onto the independent variables. The role of the log-odds in the context of logistic regression is to simply formulate the problem as a linear regression task. For a training set $\mathbb{X} = \{\boldsymbol{x}^{(i)}, y^{(i)}\}$, $y^{(i)} \in \{0, 1\}$, $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$, and $i \in \{1, ..., N\}$, the binary logistic regression can be written in the mathematical form of the Equation 9.

$$\ln \frac{Pr(y=1)}{Pr(y=0)} = \ln \frac{p}{1-p} = \boldsymbol{\beta} \cdot \boldsymbol{x} \tag{9}$$

where $p = Pr(y = 1)$ represents the probability of success for the response variable y that follows a Bernoulli distribution, $\boldsymbol{x}$ is the d-dimensional vector of independent variables, $\boldsymbol{\beta}$ is the vector of regression coefficients. The term $\ln \frac{p}{1-p}$ represents the log-odds of the outcome probability. By reordering this equation, we arrive at Equation 10,

$$p(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\beta} \cdot \boldsymbol{x}}} = sigmoid(\boldsymbol{\beta} \cdot \boldsymbol{x}) \tag{10}$$

Unlike linear least square regression, the coefficients of the logistic regression model are not calculated by closed-form expressions. The unknown regression coefficient vector $\boldsymbol{\beta}$ can be jointly found through an optimization task that aims to maximize the likelihood function $\mathcal{L}(\boldsymbol{\beta} \mid \mathbb{X})$ which is expressed in the mathematical form of Equation 11. Equivalently, the optimization can be done by minimizing the negative logarithm of the likelihood function (a.k.a. negative log-likelihood) [61]. In the case of binary logistic regression, negative log-likelihood is equivalent to binary cross-entropy loss function $L(\mathbb{X})$ which is expressed in the mathematical form of Equation 12 [61].

$$\mathcal{L}(\boldsymbol{\beta} \mid \mathbb{X}) = \prod_{i=1}^{N} y^{(i)^{p(\boldsymbol{x}^{(i)})}} (1 - y^{(i)})^{(1 - p(\boldsymbol{x}^{(i)}))} \tag{11}$$

$$\begin{aligned} L(\mathbb{X}) &= -\ln \mathcal{L}(\boldsymbol{\beta} \mid \mathbb{X}) \\ &= -\sum_{i=1}^{N} y^{(i)} \ln\Big(p(\boldsymbol{x}^{(i)})\Big) + (1 - y^{(i)}) \ln\Big(1 - p(\boldsymbol{x}^{(i)})\Big) \end{aligned} \tag{12}$$

The binary cross-entropy loss function is then minimized with respect to the parameters $\boldsymbol{\beta}$ through a numerical optimization method such as gradient descent [61]

or coordinate descent [62]. Since this loss function is convex and differentiable, a globally optimal solution can be found at the location of the global extremum.

For a response variable y that follows a multinomial distribution with K possible outcomes or classes (a.k.a multi-class classification), the logistic regression classifier can be formulated with $K-1$ independent binary logistic regression models [61]. This is the standard one-vs-all decomposition of a multi-class classification problem to several binary classification problems. In this case, the binary logistic regression model at Equation 9 can be generalized to a set of $K-1$ independent binary logistic regression models as stated in the system of equations 13.

$$\begin{aligned} \ln \frac{Pr(y=1)}{Pr(y=K)} &= \boldsymbol{B}_{1,:} \cdot \boldsymbol{x} \\ \ln \frac{Pr(y=2)}{Pr(y=K)} &= \boldsymbol{B}_{2,:} \cdot \boldsymbol{x} \\ &\dots \\ \ln \frac{Pr(y=K-1)}{Pr(y=K)} &= \boldsymbol{B}_{K-1,:} \cdot \boldsymbol{x} \end{aligned} \tag{13}$$

where $\boldsymbol{B}$ is the regression coefficient matrix with $K-1$ rows. The coefficient vector $\boldsymbol{B}_{1,:}$ refers to the first row of the coefficient matrix $\boldsymbol{B}$. This formulation can be simplified and solved for $Pr(y=K)$ as shown in Equation 14.

$$\begin{aligned} Pr(y=K) &= 1 - \sum_{m=1}^{K-1} Pr(y=m) \\ &= 1 - \sum_{m=1}^{K-1} Pr(y=K) e^{\boldsymbol{B}_{m,:} \cdot \boldsymbol{x}} \\ &= \frac{1}{1 + \sum_{m=1}^{K-1} e^{\boldsymbol{B}_{m,:} \cdot \boldsymbol{x}}} \end{aligned} \tag{14}$$

Once the value of $Pr(y=K)$ is calculated, we can derive the other probabilities as shown in the system of equations 15.

$$
\begin{aligned}
Pr(y=1) &= \frac{e^{\boldsymbol{B}_{1,:}\cdot\boldsymbol{x}}}{1+\sum_{m=1}^{K-1} e^{\boldsymbol{B}_{m,:}\cdot\boldsymbol{x}}} \\
Pr(y=2) &= \frac{e^{\boldsymbol{B}_{2,:}\cdot\boldsymbol{x}}}{1+\sum_{m=1}^{K-1} e^{\boldsymbol{B}_{m,:}\cdot\boldsymbol{x}}} \\
&\ldots \\
Pr(y=K-1) &= \frac{e^{\boldsymbol{B}_{K-1,:}\cdot\boldsymbol{x}}}{1+\sum_{m=1}^{K-1} e^{\boldsymbol{B}_{m,:}\cdot\boldsymbol{x}}}
\end{aligned}
\tag{15}
$$

It is worth noting that logistic regression can easily overfit to training data [61]. As a result, a regularization term is usually added to its loss function. Elastic-net [63], L_1-norm [61], and L_2-norm [61] are among the most popular regularization techniques.

4.2 Support Vector Machine

A binary support vector machine (SVM) is a classification technique that is designed for finding a decision boundary between two sets of samples in which the decision boundary is farthest away from the samples in each set [64; 61]. The SVM attends to this problem by defining a margin value r which is the smallest distance between the decision boundary and any of the positive and negative samples that are located on either side of the boundary [61]. The optimal decision boundary is the one that maximizes r [64; 61]. To identify such an optimal decision boundary, it is sufficient to consider only a small subset of samples in the dataset that are located near the boundary between the two classes. These samples are called the support vectors [61]. To find the optimal decision boundary, the SVM classifier has to solve an optimization problem that seeks the maximum value of r [61].

For a binary SVM and training set $\mathbb{X} = \{\boldsymbol{x}^{(i)}, y^{(i)}\}$, $y^{(i)} \in \{-1, 1\}$, $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$, and $i \in \{1, ..., N\}$, suppose that there exists a decision boundary or an algebraic hyperplane that perfectly separates the positive and negative samples [65]. This hyperplane is of the form $\boldsymbol{w} \cdot \phi(\boldsymbol{x}) + b = 0$ where $\boldsymbol{w}$ is the normal vector, b is the intercept, $\frac{|b|}{\|\boldsymbol{w}\|}$ is the perpendicular distance from the hyperplane to the origin, and $\|\boldsymbol{w}\|$ is the Euclidean norm of w, and $\phi(\boldsymbol{x})$ denotes a feature-space transformation [65]. In the case of a linear SVM, $\phi(\boldsymbol{x})$ simply represents the identity function. Given this formulation, there exists a hyperplane H_1 of the form $\boldsymbol{w} \cdot \phi(\boldsymbol{x}) + b = +1$, which passes through the support vectors in the positive class and is located at d distance from the decision boundary. The hyperplane H_1 is the lower bound of the subspace of positive class samples that is spanned by the inequality $\boldsymbol{w} \cdot \phi(\boldsymbol{x}) + b \geq +1$ for $y^{(i)} = +1$. Similarly, there exists another hyperplane H_2 of the form $\boldsymbol{w} \cdot \phi(\boldsymbol{x}) + b = -1$ which

passes through the support vectors in the negative class and is located at d distance from the decision boundary. The hyperplane H_2 is the upper bound of the subspace of negative class samples that is spanned by the inequality $\boldsymbol{w} \cdot \phi(\boldsymbol{x}) + b \leq -1$ for $y^{(i)} = -1$ which spans the subspace of the negative class samples [65]. We can combine these two inequalities into a compact form as in Equation 16, which forms a constraint to the aforementioned optimization problem that the SVM classifier has to solve when searching for the optimal values of $\boldsymbol{w}$ and b which are the parameters of the decision boundary.

$$y^{(i)}(\boldsymbol{w} \cdot \phi(\boldsymbol{x}) + b) - 1 \geq 0 \quad \forall i \tag{16}$$

By definition, the optimal decision boundary has to be located precisely in the middle of the two parallel hyperplanes H_1 and H_2. On the other hand, we know that the distances of these two hyperplanes from the origin are $\frac{1-b}{\|\boldsymbol{w}\|}$ and $\frac{-1-b}{\|\boldsymbol{w}\|}$, respectively. By subtracting these two distance values we arrive at the margin value $r = \frac{2}{\|\boldsymbol{w}\|}$. The main objective of the SVM classifier is to maximize r which naturally translates to minimization of $\|\boldsymbol{w}\|$, or for mathematical convenience, minimization of the quadratic term in 17 [65].

$$\frac{1}{2}\|\boldsymbol{w}\|^2 \tag{17}$$

To minimize the quadratic term in 17 given the constraint in 16, SVM uses a primal Lagrangian of the form presented in Equation 18 which shows the loss function we would like to minimize.

$$L(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{N} a_i y^{(i)}(\boldsymbol{w} \cdot \phi(\boldsymbol{x}^{(i)}) + b) + \sum_{i=1}^{N} a_i \tag{18}$$

Here vector $\boldsymbol{a}$ represents Lagrange multipliers [64]. Since the loss function itself is convex quadratic and the linear constraints also form a convex set, the whole Lagrangian is a convex quadratic programming problem [65] which is guaranteed to have a global minimum with respect to $\boldsymbol{w}$ and b. By taking the partial derivatives with respect to $\boldsymbol{w}$ and b and setting them to zero, and eliminating them from $L(\boldsymbol{w}, b, \boldsymbol{a})$, the dual representation of Lagrangian $\tilde{L}(\boldsymbol{a})$ appears in Equation 19 which is a function of Lagrange multipliers.

$$\tilde{L}(\boldsymbol{a}) = \sum_{i=1}^{N} a_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} a_i a_j y^{(i)} y^{(j)} \phi(\boldsymbol{x}^{(i)}) \cdot \phi(\boldsymbol{x}^{(j)}) \tag{19}$$

subject to the Karush-Kuhn-Tucker conditions [61; 65]. Equation 19 shows that the solution to the optimization problem solely depends on the pair-wise dot product of the samples or pair-wise dot product of some transformation ϕ of the samples in the training set.

At the end, for every data point, either $a_i = 0$ or $y^{(i)}(\boldsymbol{w} \cdot \phi(\boldsymbol{x}^{(i)}) + b) = 1$. The dual representation of Lagrangian facilitates finding the support vectors which are the only ones having non-zero a_i coefficients. By finding the positive and negative class support vectors we can find the hyperplanes H_1 and H_2 and accordingly the decision boundary which is located right in the middle of these two. The dual representation given in 19 has another property that facilitates the use of kernel trick. By definition, there exists a family of high-dimensional transformations $\phi : R^d \rightarrow R^z$ for $z \gg d$ that are computationally expensive and in some cases intractable when they are directly applied to input vector $\boldsymbol{x}$. However, the dot product $\langle \phi(\boldsymbol{x}^{(i)}), \phi(\boldsymbol{x}^{(j)}) \rangle$ is computationally tractable and attainable without directly computing the transformation ϕ [61]. Two of the function kernels that are widely used in the context of SVM classifiers are homogeneous polynomial and Gaussian radial basis function [65].

The formulation of SVM classifier optimization given in Equation 18 has assumed that the positive and negative samples are perfectly separable and no sample is located inside the margin area or among the opposite class samples. In order to enable the SVM classifier on tasks where there exist positive and negative samples on the wrong side of the decision boundary, we have to relax the hard margin constraint [64; 61]. To this end, we can add a slack parameter $\boldsymbol{\xi}$ to the formulation of positive and negative class subspaces to widen their spans. The resulting modified subspace formulations are $\boldsymbol{w} \cdot \phi(\boldsymbol{x}^{(i)}) + b \geq +1 - \xi_i$ for $y^{(i)} = +1$ and $\boldsymbol{w} \cdot \phi(\boldsymbol{x}^{(i)}) + b \leq -1 + \xi_i$ for $y^{(i)} = -1$ and $\xi_i \geq 0$, $\forall i$. Given the addition of slack parameter ξ_i, the Lagrangian in Equation 19 changes to the new formulation represented in Equation 20.

$$L(\boldsymbol{w}, b, \boldsymbol{a}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + c\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} a_i\{y^{(i)}(\boldsymbol{w} \cdot \phi(\boldsymbol{x}^{(i)}) + b) - 1 + \xi_i\} - \sum_{i=1}^{N} \mu_i \xi_i \quad (20)$$

where the μ_i are the multipliers introduced to enforce the positivity of the ξ_i. In addition, c is a user-defined parameter designed to control the penalty of errors with higher values enforcing a larger penalty.

For multi-class classification with K possible classes, we can follow the one-vs-all strategy as in the logistic regression classifier. Alternatively, one can construct $K(K-1)/2$ one-versus-one pairwise classifiers that are organized into a directed acyclic graph (interested readers can refer to the scikit-learn implementation of one-versus-one SVM) [61; 66]. Then, to classify a test point one has to find out which class has the highest number of votes among all the pairwise classifiers [61].

4.3 Tree-based Classifiers

In this section, the descriptions of tree-based classifiers, namely decision tree, random forest, robust boosting, and extreme gradient boosting are provided.

4.3.1 Decision Tree

The tree-based classification and regression methods build tree-like structures for determining the target variable class based on a set of input features [67]. The fundamental component of all tree-based techniques is a classifier that is called a decision tree [67]. The most well-known and used decision tree algorithm is the classification and regression trees (CART) algorithm [67]. There are other decision tree variants such as ID3 [68] and C4.5 [69]; however, we only focus on CART as it has been the only decision tree variant utilized in this thesis work.

The CART is a binary decision tree in which at every node the data is partitioned into two parts based on a yes/no question. During the training process, the first or root node of the tree is built by searching for the feature that can best partition the data. Once the root node is formed and the data have been partitioned, there are two child branches below the root node. For every child branch, the same search process is repeated to find the next best feature. To determine what feature is the best, Gini index impurity [67] is computed at each node for all the features that have not been yet used in the previous nodes of the same branch. For a K class classification problem, the Gini index can be calculated using Equation 21.

$$I_G(p) = 1 - \sum_{i=1}^{K} p_i^2 \ , \quad i \in \{1, ..., K\} \tag{21}$$

where p_i represents the fraction of items labeled into class i. For each branch, the tree growth is terminated according to some pre-defined conditions. The maximum depth of the tree, the minimum Gini index, and the minimum number of samples that ended up in a node are some of the widely used termination conditions [67; 70]. The nodes of the tree of which no child branch is created are called leaf nodes. At each leaf node, the probabilities of belonging to each of the available classes are calculated during the training process and stored [67]. In a leaf node, probabilities are simply the fraction of the samples of the same class [67]. During the test and inference process, the leaf at which an input observation will end up is picked and classified based on the stored class probabilities. The class with the highest probability is chosen as the final prediction.

4.3.2 Random Forest

A random forest classifier constructs many independent decision tree classifiers which can be created in parallel [71]. In a random forest, the final prediction probabilities are calculated by averaging the predicted probabilities of all individual decision trees per each target class. Mathematically, for a K class classification problem,

a random forest classifier can be represented by Equation 22.

$$\hat{y}_i = \frac{1}{K} \sum_{t=1}^{T} g_t(\boldsymbol{x}^{(i)}), \quad g_t \in \mathcal{G} \tag{22}$$

where $\hat{y}_i$ is the prediction probability of class i, T stands for the total number of decision trees, and g is a function in the functional space $\mathcal{G}$, and $\mathcal{G}$ is the set of all possible decision trees.

Random forest was primarily designed to mitigate the decision tree overfitting issue. In an ensemble of decision trees, i.e. weak learners, a concept known as bootstrap aggregation or bagging is implemented via the random forest. Bagging signifies that the trees of the forest are each trained on a bootstrapped sample from the training set, all of which vote for the final prediction of an input observation $\boldsymbol{x}$. It has been shown that the overall variance of the ensemble is reduced in comparison with each of the decision trees [71]. As a regularization, injecting different randomness into the tree formation and training process has been effective for improving the performance [71]. Performing random splits at each node of the trees as well as using only a random subset of available input features are two of these regularization techniques [71].

4.3.3 Adaptive Boosting

In principle, adaptive boosting (AdaBoost) works by fitting weak learners (or base learners), i.e. models that do slightly better than random guesses, to repeatedly modified versions of data [72]. Based on a weighted majority vote, all predictions will be combined [72]. The weak learners in AdaBoost are a special type of decision tree that is called decision stump. Unlike trees in a random forest, decision stumps have only one root node and two leaf nodes. AdaBoost deploys a forest of such stumps, but unlike random forest, the trees are created sequentially. To be precise, throughout this document, the term AdaBoost refers to the AdaBoost.M1 or discrete AdaBoost algorithm [72].

For a binary AdaBoost classifier, suppose we have a training set $\mathbb{X} = \{\boldsymbol{x}^{(i)}, y^{(i)}\}$, $y^{(i)} \in \{-1, 1\}$, $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$, and $i \in \{1, ..., N\}$. In an adaptive boosting framework, the first step in the training process is to produce a decision stump using the feature that delivers the lowest weighted average of the Gini index (see Equation 21) over the two leaf nodes in the stump. For each so-called boosting iteration, each training sample is modified by a set of weights $w_1, w_2, ..., w_N$. We initially give equal weight to all samples for the first stump, i.e. $w_i = \frac{1}{N}$. The sample weights are modified with each successive iteration, and the learning algorithm is then applied to the reweighted data. As the training examples are reweighted, those that were incorrectly predicted by the last decision stump are increased in weight. On the contrary,

the weights for the correctly predicted examples are decreased. With each iteration, difficult examples receive greater influence [72]. Thus, the next weak learner in the sequence is forced to concentrate on examples that the previous one has missed. The weak learners themselves are also weighted by a coefficient α that correlates with the predictive power of the stump measured over the training set.

Generally, boosting fits an additive model to the data in a forward stagewise fashion [72]. In the case of AdaBoost with a total of M decision stumps, the decision function of an additive model can be of the form

$$c(\boldsymbol{x}) = \text{sgn}\left(\sum_{m=1}^{M} \alpha_m g_m(\boldsymbol{x})\right), \quad m = 1, ..., M \tag{23}$$

The objective of an AdaBoost algorithm during the training phase is to find optimal values of α_m as well as optimal decision stumps $g_m(\boldsymbol{x})$ such that an exponential loss function $L(y, c(\boldsymbol{x})) = \exp[-yc(\boldsymbol{x})]$ is minimized. This objective can be written in the following form

$$\begin{aligned}(\alpha_m, g_m) &= \underset{\alpha, G}{\text{argmin}} \sum_{i=1}^{N} \exp\Big[-y^{(i)}(c_{m-1}(\boldsymbol{x}^{(i)}) + \frac{\alpha}{2} g(\boldsymbol{x}^{(i)}))\Big] \\ &= \underset{\alpha, G}{\text{argmin}} \sum_{i=1}^{N} w_{i,m} \exp\Big[-y^{(i)} \frac{\alpha}{2} g(\boldsymbol{x}^{(i)}))\Big]\end{aligned} \tag{24}$$

where $w_{i,m}$ is the weighting coefficient of the input observation i at the formation of decision stump m. The optimal function g_m is the one that minimizes the sum of weight values of the misclassified examples as shown in Equation 25.

$$g_m = \underset{G}{\text{argmin}} \sum_{i=1}^{N} w_{i,m} \, \mathbf{1}(y^{(i)} \neq g(\boldsymbol{x}^{(i)})) \tag{25}$$

Consequently, the optimal value of the decision stump influence coefficient α_m is calculated via Equation 26.

$$\alpha_m = \ln \frac{(1 - err_m)}{err_m} \tag{26}$$

such that the weighted error rate err_m is calculated using Equation 27.

$$err_m = \frac{\sum_{i=1}^{N} w_i \, \mathbf{1}(y^{(i)} \neq g_m(\boldsymbol{x}^{(i)}))}{\sum_{i=1}^{N} w_i} \tag{27}$$

A stump's influence on the final classification is measured by its corresponding value of α. A decision stump that does well, or has no misclassifications, has an error rate of 0 and a relatively high value of α.

The approximation for the ensemble model is then updated at iteration m via

$$c_m(\boldsymbol{x}) = c_{m-1}(\boldsymbol{x}) + \frac{\alpha}{2} g_m(\boldsymbol{x}) \tag{28}$$

The sample weights at iteration $m+1$ can be updated for every data point once the influence coefficient for each stump has been measured at the end of iteration m. This is done via the following formula

$$w_{i,m+1} = w_{i,m} \exp\Big[\alpha_m \, \mathbf{1}(y^{(i)} \neq g_m(\boldsymbol{x}^{(i)}))\Big], \quad i = 1, ..., n \tag{29}$$

where $g_m(\boldsymbol{x}^{(i)})$ is the predicted class of the last decision stump for input sample $\boldsymbol{x}^{(i)}$. Once all values of w_i are calculated, they are usually normalized so that they sum to one. If an observation is misclassified by g_m, the weights of that observation are scaled by a factor $\exp(\alpha_m)$, increasing their relative influence for the construction of the next decision stump g_{m+1}.

For multi-class classification, there are different variants of the AdaBoost classifier. Stagewise additive modeling using a multi-class exponential loss function (SAMME) [73] is the implementation that has been widely used and is implemented in Scikit-Learn library [66]. The AdaBoost-SAMME implementation for a multi-class classification requires subtle changes to the binary AdaBoost. Interested readers are encouraged to read the original paper [73] for more information on SAMME algorithm implementation.

4.3.4 Robust Boosting

Despite the unprecedented performance of AdaBoost, it was shown that the AdaBoost classifier is over-sensitive to label noise. The performance of AdaBoost was found to rapidly decrease after the training set was contaminated with label noise [74]. At every iteration of the AdaBoost algorithm, weights are increased for misclassified observations. These weights can become very large in some iterations. When this occurs, the boosting algorithm sometimes focuses on a few misclassified observations and ignores the rest of the training observations. As a result, the average classification accuracy is compromised [74]. This issue is pronounced when there is label noise in the training dataset. To overcome this issue, other variants of AdaBoost such as LogitBoost were proposed [75]. The weights placed on any example by LogitBoost are bounded, unlike Adaboost, which places unbounded weights on mislabeled examples [75]. This decreases the penalty for mislabeled examples and increases the algorithm's tolerance for noise [74]. All of AdaBoost's and LogitBoost's potential loss functions are convex [76] allowing us to calculate the loss function's minimum efficiently [74]. Random label noise, however, can still cause issues to a boosting algorithm that uses a convex loss function [77]. The problem is

caused by the inability of the optimization algorithm in AdaBoost to ignore the training examples that have label noise. These examples cannot be classified correctly because their ground truth labels were chosen mistakenly. Putting too much focus on these examples results is compromising the overall classification power [77].

AdaBoost has another weakness. Several experiments have revealed that the test error of the generated strong classifier continues to decline for many boosting iterations even after the training error converges to zero [74]. This suggests that we need to discover the criterion AdaBoost optimizes which leads to improved predictive power on the test set even after the training error converges to zero [74].

To overcome the above-mentioned shortcomings, robust boosting (RobustBoost) has been proposed which, first of all, does not assign all the weights to the misclassified observations [74] which improves the average classification accuracy. Second of all, in RobustBoost the optimization is done via a potential function that is not convex and changes as a consequence of boosting [74]. In other words, RobustBoost does not minimize the training error similarly as AdaBoost or LogitBoost does. In contrast, it maximizes the proportion of observations with the margin of classification r above a certain threshold θ.

The RobustBoost algorithm is motivated by the theory of large margins [74]. SVM classifier which was described in Section 4.2 is another popular classification technique that lends its power to maximizing the decision margin. Recalling that the decision function of an additive model is $c(\boldsymbol{x})$ defined in Equation 23, the unnormalized margin of an input observation is defined to be the product of the classification function and the predicted label [74] as shown in Equation 30.

$$r(\boldsymbol{x}, y) = y\, c(\boldsymbol{x}) \tag{30}$$

It is worth noting that the classification function $c(\boldsymbol{x})$ is a linear combination of weak learners $g_m(\boldsymbol{x})$. Naturally, observations with positive margin $r(\boldsymbol{x}, y) > 0$ are classified correctly, while those with negative margin $r(\boldsymbol{x}, y) < 0$ are misclassified. The goal of the classification algorithm is to produce as many positive margins as possible [72]. In addition, it is best not to hugely penalize the classifier for those observations that are delivering a large negative margin [74].

Given the definition of the margin $r(\boldsymbol{x}, y)$, there exists an indicator function $\mathbf{1}[r(\boldsymbol{x}, y) < 0]$ such that it holds value 1 if a sample is misclassified and 0 otherwise [74]. This indicator function is called the error step function [74]. The goal of the RobustBoost training algorithm is to minimize the total training error $E_{\mathbb{X}}$ over the training set $\mathbb{X}$ which can be expressed as

$$E_{\mathbb{X}}[c(\boldsymbol{x}) \neq y] = \frac{1}{N} \sum_{i=1}^{N} \mathbf{1}[r(\boldsymbol{x}^{(i)}, y^{(i)}) < 0] \tag{31}$$

With this formulation, the training data are linearly separable if there is a linear classifier with a training error of zero.

If we define a normalized margin function via the expression below

$$\bar{r}(\boldsymbol{x}, y) = \frac{y \operatorname{sgn}(\sum_m \alpha_m g_m(\boldsymbol{x}))}{\sum_m \mid \alpha_m \mid} \tag{32}$$

we can define the generalization error $E_D[\bar{r}(\boldsymbol{x}) \leq 0]$ [74] as the probability that $c(\boldsymbol{x}) \neq y$ when $(\boldsymbol{x}, y)$ is generated by the underlying true population distribution D [74]. Given this definition and according to the margin theory, large positive margins on training examples can lead to small generalization error [74]. In other words, to minimize the generalization error, one should find a linear classifier $c(\boldsymbol{x})$ that minimizes the number of training examples in which $\bar{r}(\boldsymbol{x}) \leq \theta$ for a large value of θ [74]. The rest of the training data has a margin greater than θ. Therefore, we need an algorithm that finds a coefficient vector $\boldsymbol{\alpha}$ such that $\bar{r}(\boldsymbol{x}) > \theta$ for most but not all of the training observations [74]. Hence, instead of minimizing the number of errors on the training set, we redefine the goal of the boosting algorithm to minimize the number of observations whose normalized margin is smaller than a positive threshold value θ [74]. Suppose we have a training set $\mathbb{X} = \{\boldsymbol{x}^{(i)}, y^{(i)}\}$, $y^{(i)} \in \{-1, 1\}$, $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$, and $i \in \{1, ..., N\}$.

Given the definition of the error step function, we can define a potential function $\Phi(r)$, which is a decreasing function of the margin $r(\boldsymbol{x}, y)$, which is an upper bound for the error step function, i.e. $\Phi(r) \geq \mathbf{1}[r(\boldsymbol{x}, y) \leq 0]$ [74]. In contrast to the error step function, the potential function $\Phi(r)$ is selected such that it is differentiable over all values of the margin. Since the potential function $\Phi(r)$ sets an upper bound on the classification error, reducing $\Phi(r)$ is a good heuristic for decreasing the classification error in boosting setups. For the sake of completeness, the exponential loss function in AdaBoost can be regarded as equivalent to the potential function in RobustBoost [74].

Gradient descent can be used to minimize the potential function $\Phi(r)$. Utilizing the chain rule, we obtain a simple expression for the derivative w.r.t. α_m at every boosting iteration m. Denoting $r(\boldsymbol{x}^{(i)}, y^{(i)})$ by r_i we get

$$\begin{aligned}
\frac{\partial}{\partial \alpha_m} \frac{1}{N} \sum_{i=1}^{N} \Phi(r_i) &= \frac{1}{N} \sum_{i=1}^{N} \frac{\partial r}{\partial \alpha_m} \frac{\partial \Phi(r)}{\partial r} \bigg|_{r=r_i} \\
&= \frac{1}{N} \sum_{i=1}^{N} y^{(i)} g_m(\boldsymbol{x}^{(i)}) \frac{\partial \Phi(r)}{\partial r} \bigg|_{r=r_i} \\
&= -\frac{1}{N} \sum_{i=1}^{N} y^{(i)} g_m(\boldsymbol{x}^{(i)}) w(r_i)
\end{aligned} \tag{33}$$

$$\frac{\partial \Phi(r)}{\partial r} = -w(r) \tag{34}$$

Based on Equation 33, the derivative of the potential function w.r.t. α_m is equal to the correlation between ground truth labels and the classifier predictions weighted by the coefficient $-w(r_i)$. The weighting coefficient is simply equal to the minus derivative of the potential with respect to the margin as shown in Equation 34. Such a formulation allows us to flexibly control the contribution of each training observation on the value of the potential function. If the goal is to ignore the training observations with excessively large margins, we can choose a potential function such that its partial derivative with respect to the margin r decreases as the magnitude of the margin grows beyond the threshold defined by θ.

As mentioned before, the RobustBoost algorithm aims at indirectly minimizing the training error via the potential function. RobustBoost training is based on time evolution denoted by the parameter t and not the training error. At every step of the algorithm, RobustBoost searches for a positive step in time Δt and a positive change in the average margin of the training data Δr. The parameter t ($0 \leq t \leq 1$) controls the progress of the training process and its termination. The potential function $\Phi(r, t)$ is then defined as

$$\Phi(r, t) = 1 - err(s(r, t)) \tag{35}$$

where the error function err is defined as

$$err(a) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{a} \exp\left(-x^2\right) dx \tag{36}$$

and the function $s(r, t)$ performs scaling and translation utilizing the hyper-parameters of the algorithm and the time variable t. The optimal values of Δr and Δt are found via an optimization process. Please refer to the original publication for more details on the function $s(r, t)$ as well as the hyper-parameters [74].

Taking the partial derivative of $\Phi(r, t)$ with respect to the margin r we should get a bell-shaped curve which nicely satisfies the property of dampening the effect of training observations with excessively large margins.

For multi-class classification, one can follow the one-vs-all approach as in logistic regression.

4.3.5 Gradient-Boosted Trees

A gradient-boosted trees algorithm is a machine learning technique that is used for both classification and regression tasks. It works by combining many weak learners, i.e. regression trees, into a stronger predictor [78; 72]. Similar to other boosting methods, the main idea behind the gradient boosting algorithm is to create weak

learners sequentially in a way that in each step the new learners try to compensate for the errors of the previous learners [72]. The weak learners are always regression trees for both regression and classification tasks in gradient boosted trees because the algorithm is designed to optimize the pseudo-residuals of predictions iteratively [78]. Since gradient boosting is a form of gradient descent in function space [79; 80], the gradients of the loss with respect to the candidate weak learner functions relate directly to the pseudo-residuals of the whole ensemble model. This means that by moving toward the opposite direction of the gradients returned by the gradient boosting algorithm, the algorithm is reducing the pseudo-residuals of the ensemble model; consequently, achieving a better fit [79; 80]. This is the central property and the main reason behind the success of the gradient boosting algorithm.

In each iteration of gradient boosting, a new weak learner is formed greedily very much similar to the random forest, and then altered in such a way that it reduces the pseudo-residuals of the predictions made by the previously formed weak learners. The alterations happen at the leaf nodes of the regression trees and are controlled by the gradients of the loss function. The loss function can be any arbitrary differentiable function $L(y, g(\boldsymbol{x}))$ that allows characterizing the prediction error of the classification model.

In a gradient boosting framework, the goal is to optimize the classification error iteratively. Similar to the logistic regression classifier, the gradient-boosted trees algorithm works with log-odds of the predicted probabilities. In other words, the output at each regression tree leaf is expressed in terms of log-odds. Subsequently, the loss function measures the classification error in terms of the log-odds, very much similar to logistic regression. In the context of gradient-boosted trees, the negative of the loss function gradients with respect to the latest ensemble model represents the pseudo-residuals [78]. Naturally, to minimize the pseudo-residuals, the learner needs to move toward the opposite direction of the gradients for each training observation [78]. This is done via sliding the predicted log-odds of each regression tree leaf node toward the opposite direction of the gradients [72]. It is worth noting that when regression trees are used as base learners, the input observations are grouped into disjoint regions. As a result, the optimization of the ensemble model cannot be done per each observation, but rather, per each disjoint region. This is the main reason for calling this method a stochastic gradient boosting [78]. This process is described in the following paragraphs.

For a training set $\mathbb{X} = \{\boldsymbol{x}^{(i)}, y^{(i)}\}$, $y^{(i)} \in \{0, 1\}$, $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$, and $i \in \{1, ..., N\}$, the classifier first needs a differentiable loss function $L(y^{(i)}, g(\boldsymbol{x}^{(i)}))$. In the first iteration $m = 0$, the gradient boosted trees algorithm makes a rough estimate of the log-odds value γ that minimizes the binary cross-entropy loss function $L(y^{(i)}, \gamma)$ which was introduced in Equation 12. In mathematical form, this can be expressed

as

$$g_0(\boldsymbol{x}) = \gamma = \underset{\gamma}{\operatorname{argmin}} \sum_{i=1}^{N} L(y^{(i)}, \gamma) \tag{37}$$

where function g_m refers to the output value of the ensemble model at iteration m. The optimal value of γ at the very first iteration $m = 0$ can be obtained by a closed-form solution by taking the derivative of the binary cross-entropy loss function with respect to γ. The optimal value of γ here turns out to be the ratio of the number of positive class observations over the negative class observations in the training set. The value of γ is then used as the initial output of the ensemble model which is then used to measure the pseudo-residuals of the classification per each training observation. The pseudo-residuals $\tilde{y}_m^{(i)}$ at each iteration m are simply the negative of the loss function gradients with respect to the current model g_{m-1} expressed as

$$\tilde{y}_m^{(i)} = - \left[\frac{\partial L(y^{(i)}, g(\boldsymbol{x}^{(i)}))}{\partial g(\boldsymbol{x}^{(i)})} \right]_{g(\boldsymbol{x})=g_{m-1}(\boldsymbol{x})} \tag{38}$$

In the next iterations $m \geq 1$, regression trees are formed one after another. The created weak learners are forced to predict the pseudo-residuals of the ensemble model at the previous iteration. At the end of each iteration, new pseudo-residuals are obtained per training observation. At this point, the training algorithm aims at tweaking the output of each leaf node $\gamma_{j,m}$ indexed by $j = 0, ..., J$ such that the average loss for all samples $\boldsymbol{x}^{(i)}$ that ended up in the leaf node is decreased and likewise the average amount of pseudo-residuals. If each leaf node is represented by $R_{j,m}$, then this process can be expressed as

$$\gamma_{j,m} = \underset{\gamma}{\operatorname{argmin}} \sum_{\boldsymbol{x}^{(i)} \in R_{j,m}} L(y^{(i)}, g_{m-1} + \gamma) \tag{39}$$

where $g_{m-1}(\boldsymbol{x}^{(i)})$ is the output of the whole ensemble model at the previous iteration for the input sample $\boldsymbol{x}^{(i)}$. In other words, the optimal value of $\gamma_{j,m}$ at leaf node R_j at iteration m is the one that minimizes the given summation in Equation 39. The optimal value of $\gamma_{j,m}$ can be approximated by a closed form solution via obtaining the second order Taylor expansion of the function $L(y^{(i)}, g_{m-1} + \gamma)$ near the point $g_{m-1}(\boldsymbol{x}^{(i)})$ [81; 72] as shown in Equation 40.

$$L(y^{(i)}, g_{m-1} + \gamma) \approx L(y^{(i)}, g_{m-1}) + \frac{\partial L(y^{(i)}, g_{m-1})\gamma}{\partial g_{m-1}} + \frac{1}{2}\frac{\partial^2 L(y^{(i)}, g_{m-1})\gamma^2}{\partial g_{m-1}^2} \tag{40}$$

where the first order and the second order partial derivative of the loss with respect to the latest predictions of the model g_{m-1} are used for approximation [81; 72]. Next, by taking the derivative with respect to γ and setting the derivative to zero we can solve for the optimal value of γ [75]. The optimal value of γ is the one that when it is summed up with the available ensemble model, the pseudo-residuals of the predictions are decreased the most. This can be expressed mathematically via

$$\gamma_{j,m} = -\frac{\sum_{\boldsymbol{x}^{(i)} \in R_{j,m}} \frac{\partial L}{\partial g_{m-1}}}{\sum_{\boldsymbol{x}^{(i)} \in R_{j,m}} \frac{\partial^2 L}{\partial g_{m-1}^2}} \tag{41}$$

Once the optimal γ is identified, it is plugged into Equation 42 which updates the output of the ensemble model. This can be expressed in mathematical form as

$$g_m(\boldsymbol{x}^{(i)}) = g_{m-1}(\boldsymbol{x}^{(i)}) + \nu \sum_{j=1}^{J_m} \gamma_m \mathbf{1}(\boldsymbol{x}^{(i)} \in R_{j,m}) \tag{42}$$

where ν represents the learning rate and the indicator vector $\mathbf{1}(\boldsymbol{x}^{(i)} \in R_{j,m})$ controls how the updates are applied to the inputs depending on the leaf node(s) they ended up in [72].

For a response variable y that follows a multinomial distribution with K possible classes, K trees are built at iteration m, one per each class [72]. Subsequently, the probability that $\boldsymbol{x}^{(i)}$ belongs to class $k \in 1, ..., K$ is modeled as a softmax of the $g_{m,k}(\boldsymbol{x}^{(i)})$ values as shown in Equation 43.

$$p_k(\boldsymbol{x}^{(i)}) = \frac{\exp\big(g_{m,k}(\boldsymbol{x}^{(i)})\big)}{\sum_{l=1}^{K} \exp\big(g_{m,l}(\boldsymbol{x}^{(i)})\big)} \tag{43}$$

Naturally, a multinomial cross-entropy loss is chosen as the loss function [72]. Accordingly, since we have K trees built at each iteration m, we need to calculate K pseudo-residuals as shown in Equation 44.

$$\begin{aligned}\tilde{y}_{i,m,k} &= -\left[\frac{\partial L(y^{(i)}, g_1(\boldsymbol{x}^{(i)})), ..., g_K(\boldsymbol{x}^{(i)}))}{\partial g_k(\boldsymbol{x}^{(i)})}\right]_{g(\boldsymbol{x})=g_{m-1}(\boldsymbol{x})} \\ &= r_{i,k} - p_k(\boldsymbol{x}^{(i)})\end{aligned} \tag{44}$$

where $r_{i,k} = 1$ if $y^{(i)} = k$ and $r_{i,k} = 0$ otherwise. Therefore, applying the gradient descent to each of the K trees exclusively with respect to the loss value of class k.

4.3.6 Extreme Gradient Boosting

Extreme gradient boosting (XGBoost) is an improvement to gradient-boosted trees with the additional regularization term to the loss function and the significantly scalable training algorithm [82]. Very much similar to gradient boosted trees, XGBoost is an additive ensemble model that builds decision tree models (CART) greedily and then alters the outputs of each tree at leaf nodes by utilizing gradients of the loss function. Each tree targets to predict the residuals of the model by altering the leaf node output values toward the opposite direction (negation) of the loss function gradients [82]. The decision trees in XGBoost are, however, formed differently. Instead of using the Gini index to determine the best splits, XGBoost uses a quality score and a gain score [81; 82]. Another difference between XGBoost and gradient-boosted trees is in how XGBoost handles boosting for a large amount of data [82]. In particular, datasets that do not fit into the random-access memory of a single computer are handled efficiently. Moreover, XGBoost training can be parallelized and distributed; hence, accelerating the training process [82]. Lastly, similar to gradient-boosted trees, XGBoost works with log-odds of the predictions.

The training algorithm in XGBoost tries to iteratively add weak learners to a model that was initialized with a global prediction probability $g_0(\boldsymbol{x})$. Suppose we have a training set $\mathbb{X} = \{\boldsymbol{x}^{(i)}, y^{(i)}\}$, $y^{(i)} \in \{0, 1\}$, $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$, and $i \in \{1, ..., N\}$. For a binary classification problem, $g_0(\boldsymbol{x})$ can be either set to the average probability of observing the positive class or simply to 0.5. In XGBoost, the objective is to find the optimal value of γ that minimizes the regularized loss function. This can be expressed as follows

$$\gamma_{j,m} = \underset{\gamma}{\operatorname{argmin}} \sum_{\boldsymbol{x}^{(i)} \in R_{j,m}} L(y^{(i)}, g_{m-1} + \gamma) + \Omega(\gamma) \tag{45}$$

where $R_{j,m}$ represents the set of data points that have landed in the node j of the

decision tree at iteration m. The regularization term $\Omega(\gamma)$ is defined as follows

$$\Omega(\gamma) = \delta T + \frac{1}{2}\sum_{j=1}^{T}\gamma_j^2 \tag{46}$$

Similar to gradient boosting, the second-order Taylor expansion of this objective can be used to quickly find the optimal values of $\gamma_{j,m}$. Simplifying and reordering the second-order approximation results in a new loss function that depends only on the first and the second partial derivatives of the loss function L and the regularization parameter λ [82]. This approximated objective can be expressed as

$$\gamma_{j,m} = -\frac{\sum_{\boldsymbol{x}^{(i)}\in R_{j,m}} \dfrac{\partial L}{\partial g_{m-1}}}{\sum_{\boldsymbol{x}^{(i)}\in R_{j,m}} \dfrac{\partial^2 L}{\partial g_{m-1}^2} + \lambda} \tag{47}$$

where the first-order and the second-order partial derivatives of the loss with respect to the latest predictions of the model g_{m-1} are used for approximation [82]. The regularization parameter λ is the L_2-norm penalty coefficient of the output values γ_j for a tree with T number of leaf nodes. Accordingly, the quality score function which helps quantify the quality of the tree structure $Quality(q)$ can be computed by

$$Quality(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{\left(\sum_{\boldsymbol{x}^{(i)}\in R_{j,m}} \dfrac{\partial L}{\partial g_{m-1}}\right)^2}{\sum_{\boldsymbol{x}^{(i)}\in R_{j,m}} \dfrac{\partial^2 L}{\partial g_{m-1}^2} + \lambda} + \delta T \tag{48}$$

Practically, it is impossible to enumerate every possible tree structure q. Instead, a greedy algorithm is used, which adds branches iteratively to the tree starting from a single leaf and finding the best split by computing a gain score $Gain(s)$ for l possible candidate splits $s \in \{s_1, s_2, ..., s_l\}$. To this end, one approximate quality score is measured per each left $\tilde{Q}_L$ and right $\tilde{Q}_R$ nodes for a candidate split and then compared with the quality score of the parent node $\tilde{Q}$. Hence, the process of split finding needs to compute $\tilde{Q}_P$, $\tilde{Q}_L$, $\tilde{Q}_R$ and then combine these into $Gain(s)$. In mathematical form, these can be expressed as

$$\tilde{Q}_{j,m} = \left.\frac{\left(\sum_{\boldsymbol{x}^{(i)}\in R_{j,m}} \dfrac{\partial L}{\partial g_{m-1}}\right)^2}{\sum_{\boldsymbol{x}^{(i)}\in R_{j,m}} \dfrac{\partial^2 L}{\partial g_{m-1}^2} + \lambda}\right|_{j\in\{L,R,P\}} \tag{49}$$

$$Gain(s) = \frac{1}{2}\left[\tilde{Q}_L + \tilde{Q}_R - \tilde{Q}_P\right] - \delta \tag{50}$$

Unlike the conventional decision trees which use a greedy split finding, candidate splits in XGBoost are found via an approximate greedy algorithm [82]. This technique relies on measuring statistics over buckets of input features formed according to the quantiles they fall into [82]. If the input dataset is too large to fit into the memory, XGBoost approximates the quantiles by approximating the statistics of the probability distribution of each input feature [82]. XGBoost makes the quantile formation more intelligent by weighting the input observations by a coefficient such that the observations that have not been confidently classified fall into separate bins [82]. The authors of XGBoost call this technique a weighted quantile sketch. This technique helps create more powerful trees and therefore faster convergence as well as better generalization [82].

Similar to other ensemble techniques such as random forest or gradient-boosted trees, the regression trees' depth and number of samples in each leaf node can be limited to build smaller trees and therefore hinder overfitting. In addition, XGBoost adopts an additional pruning technique that shrinks the decision trees after they are built by dropping the leaf nodes that have a gain score less than the user-defined regularization parameter δ [82].

The detailed description of parallel and out-of-core computation techniques of XGBoost can be found in the original publication [82]. Similarly, the description of the Sparsity-aware Split Finding technique which is solely designed to handle missing data during training can be found in the original publication [82].

For a response variable y that follows a multinomial distribution with K possible classes, K trees are built at iteration m, one per each class. The procedure is the same as what has been described for gradient boosting.

4.4 Artificial Neural Networks

Artificial neural networks (ANNs) are a family of machine learning models that are widely used for both regression and classification tasks by forming a series of nested linear and nonlinear transformations over the data [83]. Inside an ANN, there can be many perceptrons [84] or nodes that are grouped into layers. These layers of nodes can each perform any arbitrary linear transformation that is followed by any arbitrary non-linear transformation as long as they are differentiable over their input domain [83]. In supervised classification context, the goal of an ANN is to approximate a function f^* that maps input vectors $\boldsymbol{x}$ to output vectors $\boldsymbol{y}$ given a set of parameters $\boldsymbol{\theta}$. The approximated mapping $\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{\theta})$ is refined iteratively via a numerical optimization algorithm [83]. The layered structure of an ANN can be represented by a chain of functions, each function symbolizing the transformations that

happen by each layer. As an example, for a 3-layer ANN, the chain of functions in Equation 51 symbolically shows how the input is transformed and passed from layer 1 all the way to layer 3.

$$\hat{\boldsymbol{y}} = f(\boldsymbol{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\boldsymbol{x}))) \tag{51}$$

During the training process of an ANN, for each input vector, $\boldsymbol{x}^{(i)}$ the output layer must produce a vector $\hat{\boldsymbol{y}}^{(i)}$ close to the ground truth vector $\boldsymbol{y}^{(i)}$ [83]. The parameters of the layers of an ANN are usually randomly initialized and then updated iteratively by an optimization algorithm such as stochastic gradient descent (SGD) [83]. At the heart of the SGD algorithm, is the chain rule that allows measuring the magnitude and direction of the updates of the parameters through the nested composition of ANN layers [83]. In other words, the chain rule enables backpropagation of the error which is obtained at the output layer all the way to the hidden layers. Assuming that we want to find the gradients of the differentiable loss function $L(\boldsymbol{y}, \hat{\boldsymbol{y}})$ with respect to a scalar learnable parameter θ in the very first layer of the network, the chain rule can be represented symbolically via Equation 52.

$$\frac{\partial f}{\partial \theta} = \frac{\partial f^{(3)}}{\partial f^{(2)}} \frac{\partial f^{(2)}}{\partial f^{(1)}} \frac{\partial f^{(1)}}{\partial \theta} \tag{52}$$

The total number of layers or the sequential transformations determines the depth of the network. During the learning process, the training data do not specify what every layer should do, so the learning algorithm must determine how to utilize them the best to implement an approximation of f^* [83]. Since the training data does not demonstrate the desired output for each of these layers, they are referred to as hidden layers [83]. In addition to the depth, an ANN has a width property which is determined by the dimensionality of its hidden layers. Hidden layers are usually vector-valued where each element is a node and can be compared to a biological neuron [83].

For an ANN that only has two hidden layers, with any arbitrary number of nodes in the first layer and an activation function in the second layer, the mathematical transformation of the input vector $\boldsymbol{x}$ can be expressed via

$$\begin{aligned} f^{(1)}(\boldsymbol{x}; \boldsymbol{W}, \boldsymbol{b}) &= \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b} \\ f^{(2)} &= \sigma(f^{(1)}) \end{aligned} \tag{53}$$

where $f^{(1)}$ is the linear transformation (or the dot product) of the input feature vector $\boldsymbol{x}$ via the weight matrix $\boldsymbol{W}$ shifted by bias term $\boldsymbol{b}$. In addition, $f^{(2)}$ is the non-linear transformation of $f^{(1)}$ via Sigmoid function σ [83]. These types of ANNs which employ layers of perceptrons that are connected to all the input elements are called multilayer perceptron (MLP) [83]. MLPs are the oldest known ANNs and

are still very popular and effective for various classification and regression problems [83].

There exist other types of ANNs that perform the transformation operation differently when compared with Equation 53. In the following sub-sections, convolutional and recurrent neural networks which have been used throughout this thesis work are briefly described.

4.4.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) [85] are among the most popular variants of ANNs which by definition are specialized neural networks for analyzing data whose topology resembles a grid [83]. In machine learning terminology, convolution is a type of linear transformation that is applied sequentially to smaller areas of the input data grid and places the results in an output data grid. In a convolution operation, the transformations are usually applied locally with a specific order according to the relative location of the data points on the input data grid [83]. If we assume that we are dealing with a multi-channel (or multi-variate) signal $\boldsymbol{X}$ that contains C channels and has time duration $J + M - 1$, a convolutional kernel $\boldsymbol{K}$ iteratively transforms a time-shifted receptive field from $\boldsymbol{X}$. It is worth noting that $\boldsymbol{K}$ must contain the same number of channels as in the input signal. This operation is known as 1-dimensional convolution and can be expressed mathematically via

$$\boldsymbol{s}_j = grandsum(\boldsymbol{X}_{j:j+M,:} \odot \boldsymbol{K}) \,, \quad j = 0, ..., J-1 \tag{54}$$

where kernel $\boldsymbol{K}$ is a matrix of M rows and C columns which holds the transformation coefficients, very much equivalent to the weight matrix $\boldsymbol{W}$ in an MLP. The operation in Equation 54 indicates that for every data point $\boldsymbol{s}_j$ in the output, the transformation is computed by performing first a Hadamard product of a local region with a length M from the input $\boldsymbol{X}$ and the kernel $\boldsymbol{K}$ and then computing the $grandsum$ – which is the sum of all elements of the product. It can be seen that the kernel $\boldsymbol{K}$ sweeps along the time axis of the input to generate the output data points one at a time. Note that the indexing starts from 0.

It is worth noting that the transformation presented as a convolution here is similar to a cross-correlation in digital signal processing [83]. However, in the machine learning community, this transformation is known as convolution.

The formulation of the convolution operation as in Equation 54 can be easily expanded to higher dimensional arrays or tensors [83]. It is worth noting that there can be many different variants of convolution operation depending on how they compute and combine the transformations. Interested readers are encouraged to see [86].

The popularity and strength of CNNs originate from three important properties of convolution operation which are namely, sparse connectivity, parameter sharing, and equivariance to translation [83]. Sparse connectivity refers to the property that

allows a convolutional kernel to have a smaller number of elements in each dimension than the input tensor that is applied to. This property helps reduce the number of parameters needed for creating a transformed representation of an input [83]. Parameter sharing refers to the ability to sweep a small kernel over a large input tensor as opposed to the dense connection in MLP which requires one unique weight value per each input data sample [83]. When sparse connectivity and parameter sharing properties are combined, significant improvements in terms of memory consumption and network size reduction are achieved [83]. Lastly, equivariance to translation signifies a property that enables convolutional transformation to return an identical output irrespective of shifts in the input or the relative location of the receptive field over which the convolutional kernels are applied [83].

The training of a convolutional layer through back-propagation is very similar to how it is done in an MLP but with a subtle difference. Since each convolutional kernel sweeps along one or more axes of the input data grid (weight sharing), the individual coefficients inside the kernel contribute to the transformation of many input samples. In this case, when back-propagation is performed, the training algorithm has to accumulate the gradients of the loss for all the input data samples which were transformed by each individual coefficient in a kernel [83; 87].

4.4.2 Recurrent Neural Networks

Another popular variant of ANNs is the recurrent neural networks (RNNs) [88] which are specialized at processing sequential data [83]. These networks are especially good at modeling the data representations that evolve as a function of time or any equivalent independent variable. In the classical form of a dynamical or recurrent system, the state of the system at time instance t is measured by applying a function f to the previous state given the parameters of the system [83]. For example, for a process that evolves sequentially from time steps 1 to 3, the dynamical system can be represented symbolically via Equation 55.

$$\begin{aligned} \boldsymbol{s}^{(3)} &= f(\boldsymbol{s}^{(2)};\boldsymbol{\theta}) \\ &= f(f(\boldsymbol{s}^{(1)};\boldsymbol{\theta});\boldsymbol{\theta}) \end{aligned} \tag{55}$$

By definition, an RNN is a dynamical system that is capable of accounting for an input signal $\boldsymbol{x}_t$ at each time point t in addition to the past state $f(\boldsymbol{s}^{(1)};\boldsymbol{\theta})$. Such a system can be expressed in the mathematical form of Equation 56.

$$\boldsymbol{h}^{(t)} = f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)};\boldsymbol{\theta}) \tag{56}$$

where we denote the state of the system via $\boldsymbol{h}^{(t)}$ to emphasize that this is a hidden state. The hidden state $\boldsymbol{h}^{(t)}$ serves as a summary of the task-relevant features of the

past sequence of inputs up to time point t. Since $\boldsymbol{h}^{(t)}$ is a vector-valued variable of limited length, the maximum amount of information that it contains is limited. As a result, the formation of the vector $\boldsymbol{h}^{(t)}$ is in general a lossy operation [83].

With the formulation of an RNN as in Equation 56, the time duration over which the dynamical system is applied is determined by the sequence length T. However, regardless of the magnitude of T, a single function f parametrized with a fixed set of parameters $\boldsymbol{\theta}$ is reused at each time step. An RNN can model sequences of any length by learning a single shared model f. In other words, for any value of T, the RNN iteratively applies a single transformation function f to each time point t. Hence, fewer training examples are required to estimate the function f as we do not have to train with all possible sequences of arbitrary length T [83].

At each time point t transformation that happens in a vanilla RNN layer, which is composed of many RNN nodes, can be represented via

$$\boldsymbol{h}^{(t)} = tanh(\boldsymbol{b} + \boldsymbol{W}\boldsymbol{x}^{(t)} + \boldsymbol{U}\boldsymbol{h}^{(t-1)}) \tag{57}$$

where $\boldsymbol{W}$ and $\boldsymbol{U}$ are the weight matrices which transform the input vector $\boldsymbol{x}^{(t)}$ and the hidden state vector from the previous time step $\boldsymbol{h}^{(t-1)}$. The bias $\boldsymbol{b}$ is the intercept of the transformation before the hyperbolic tangent activation. Note that since in an RNN layer there can be more than one single RNN node, the hidden state and bias are vector-valued. This requires also defining the transformation weights in matrix form.

The vanilla RNN layer as represented in Equation 57 has a few shortcomings. Vanishing and exploding gradients are the two weaknesses of vanilla RNN [83]. In addition, when dealing with long sequences that are made of sub-sequences, vanilla RNN nodes are incapable of memorizing the critical information or discarding the no longer needed information [83]. To overcome these weaknesses, gated RNNs were introduced and are used effectively in practical applications [83].

Long short-term memory (LSTM) is one of the most popular variants in the family of gated RNNs [83]. In an LSTM node, there is a new quantity named cell or memory state which facilitates the passage of information from previous time points to the future. During the backpropagation, the path through the cell state allows smoother passage of gradients which in turn alleviates the vanishing gradient issue [83].

In this paragraph and the following equations, similar vector and matrix notations as in Equation 56 are used to show that an LSTM layer can contain many nodes. In an LSTM node, there is a forget gate $\boldsymbol{f}^{(t)}$ that controls the memory of the LSTM node at each time point t. The forget gate is computed via Equation 58. Once the forget gate is computed, we can compute the value of the new cell state $\boldsymbol{c}^{(t)}$. To measure the new cell state, we need to compute the input gate $\boldsymbol{i}^{(t)}$ and candidate cell state values $\tilde{\boldsymbol{c}}^{(t)}$. These input gate and candidate cell states are measured via Equation 59

and Equation 60. The new cell state value is computed via Equation 61. Finally, the new hidden state $\boldsymbol{h}^{(t)}$ is computed by combining the new cell state $\boldsymbol{c}^{(t)}$ and the output gate $\boldsymbol{o}^{(t)}$. The output gate $\boldsymbol{o}^{(t)}$ is computed via Equation 62 and the new hidden state $\boldsymbol{h}^{(t)}$ via Equation 63 [83].

$$\boldsymbol{f}^{(t)} = \sigma(\boldsymbol{b}_f + \boldsymbol{W}_f \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_f \boldsymbol{x}^{(t)}) \tag{58}$$

$$\boldsymbol{i}^{(t)} = \sigma(\boldsymbol{b}_i + \boldsymbol{W}_i \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_i \boldsymbol{x}^{(t)}) \tag{59}$$

$$\tilde{\boldsymbol{c}}^{(t)} = tanh(\boldsymbol{b}_c + \boldsymbol{W}_c \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_c \boldsymbol{x}^{(t)}) \tag{60}$$

$$\boldsymbol{c}^{(t)} = \boldsymbol{f}^{(t)} \odot \boldsymbol{c}^{(t-1)} + \boldsymbol{i}^{(t)} \odot \tilde{\boldsymbol{c}}^{(t)} \tag{61}$$

$$\boldsymbol{o}^{(t)} = \sigma(\boldsymbol{b}_o + \boldsymbol{W}_o \boldsymbol{h}^{(t-1)} + \boldsymbol{U}_o \boldsymbol{x}^{(t)}) \tag{62}$$

$$\boldsymbol{h}^{(t)} = \boldsymbol{o}^{(t)} \odot tanh(\boldsymbol{c}^{(t)}) \tag{63}$$

The training process of RNNs happens via back-propagation through time technique [83]. Assuming that we have a sequence-to-sequence (seq2seq) RNN layer, the total loss of the layer with respect to the trainable parameters is equal to the average of all output loss values from time point 1 all the way to T. Since there is a recurrent connection with shared weights inside each RNN node, the gradient of the shared weights has to be measured differently than the non-recurrent neural network weights [89]. In simple terms, the gradient of the loss with respect to shared weights has to be measured for all time steps and then aggregated [89]. More details about this aggregation can be found in [89].

4.5 Validation and Testing Approaches

Validation and testing are essential components of a machine learning project as they play a crucial role in assessing model performance on unseen data. Usually, this involves splitting the available data into three subsets: a training set to fit the model, a validation set to closely track the goodness of fit during training, and a separate test set to evaluate its prediction performance. In disciplines such as medical sciences and analogous fields, splitting must be executed in a manner that ensures the exclusive allocation of each patient's data to one of the aforementioned sets. This imperative arises from the fundamental objective of deploying a trained machine learning model to unseen individuals' data. This principle is also applied to cross-validation methods described in the following paragraphs.

When the size of the dataset is not large enough, machine learning practitioners choose to do the validation and testing differently. In this case, cross-validation is selected instead [90]. This involves splitting the dataset on the fly. That is, dividing the dataset into multiple subsets or folds, and using each fold as a validation set and the remaining as a new training set. This process is repeated for each fold, and the average prediction performance across all folds is used as an estimate of the model's generalization performance. The primary objective of validation and cross-validation is to address the issue of overfitting, where a model is excessively tailored to the training data and fails to perform well on new, unseen data. By evaluating the model on various validation sets, these techniques provide a more realistic estimation of the model's ability to generalize to new data. It is worth noting that several types of cross-validation methods exist, including k-fold cross-validation [90], leave-one-out cross-validation [91], and nested cross-validation [92]. The choice of the appropriate technique depends on factors like the dataset size and characteristics and the specific requirements of the modeling problem.

K-fold cross-validation involves dividing the data into k equal or nearly equal folds, using each fold once as a validation set, and the remaining k-1 folds as a training set [90]. The model's prediction performance is computed for each fold and averaged across all folds. While k-fold cross-validation is a widely used and straightforward technique applicable to any data set, the selection of k can influence the prediction performance estimate's variance and bias. A small k may lead to high variance as the model is evaluated on a small validation set that may not fully represent the entire data, whereas a large k may reduce variance but introduce high bias, as the model is trained on a smaller training set potentially missing important patterns. A common guideline is to use k=10, as it strikes a good balance between variance and bias.

Leave-one-out cross-validation (LOOCV) represents a special case of k-fold cross-validation where k is set equal to the number of samples in the data set [91]. Each sample is used once as a validation set, while the rest are employed as a training set. LOOCV possesses advantages and disadvantages compared to k-fold cross-validation. On the positive side, LOOCV utilizes all data for training and testing, making it low-bias and highly efficient. However, a drawback is that it can be computationally demanding, especially for large data sets, and may suffer from high variance, as the model is assessed on a single sample that might not be fully representative of the entire data.

Nested cross-validation merges cross-validation with hyperparameter tuning [93], where hyperparameters need manual specification, such as the regularization strength or the number of hidden units in a neural network. The goal is to find the optimal values for these parameters that maximize the model's performance on a validation set. However, using the same validation set for both hyperparameter tuning and model evaluation may lead to overfitting, as the model is tuned specifically for

that particular set of data. Nested cross-validation addresses this issue by employing two levels of cross-validation: an inner level for hyperparameter tuning and an outer level for model evaluation [93]. The data is initially split into outer folds, with each outer fold serving as a test set, while the rest form an inner data set. The inner data set is then subdivided into inner folds, using each inner fold as a validation set and the remaining as a training set. Hyperparameters are tuned by cross-validation on the inner folds, selecting values that maximize average prediction performance. The model with these optimal hyperparameters is then trained on the entire inner data set and evaluated on the outer test set. The prediction performance of the model is computed for each outer fold and averaged across all outer folds. This approach provides a more robust estimation of the model's performance on new data, as it avoids overfitting on any single validation set [93].

In this thesis work, different types of validation and cross-validation were exploited based on the dataset size and the nature of the problem at hand. Interested readers are encouraged to see the individual papers for more information.

5 Overview of Original Publications

5.1 Paper I: Comprehensive analysis of cardiogenic vibrations for automated detection of atrial fibrillation using smartphone mechanocardiograms

Objectives

In [25], our main objective was the classification of AFib using SCG and GCG signals via feature engineering and supervised machine learning. The data were solely collected by a smartphone built-in accelerometer and gyroscope sensors.

Approach

The MODE-AF study dataset which consists of SCG and GCG signals collected from a sample size of 300 clinical patients was used in this study. This dataset contains a curated population of elderly adults with and without AFib. In detail, MODE-AF contains 150 patients with AFib and 150 patients with sinus rhythm (SR) who were enrolled in the cardiology and internal medicine wards of Turku University Hospital, Finland, between April and September 2017. The subjects were instructed to participate in a 3-minute joint SCG and GCG recording with a Sony Xperia smartphone placed on their sternum following the obtaining of informed consent. Simultaneously, a five-lead telemetry ECG recording was recorded to determine the rhythm, supraventricular extrasystoles, and ventricular extrasystoles. The rhythm classification and interpretation of the ECG were done by two independent cardiologists. In the case of disagreement in the interpretations, a third cardiologist made the final decision. Additionally, physical measurements were recorded and electronic patient records were searched for information regarding the subjects' clinical history and investigations conducted during the index hospitalization.

We implemented (1) singular spectrum analysis and signal envelope for signal enhancement, (2) multi-disciplinary features for signal characterization, and (3) a majority voting classifier for increasing the robustness of the classification. We tested three classifiers namely, SVM, random forest, and robust boosting. These classifiers were grouped to form a majority voting classifier.

For the evaluation of the classification performance, we performed training and leave-one-person-out cross-validation on the MODE-AF dataset. Next, we trained

the classifiers on the whole MODE-AF dataset and then tested the classifiers on a separate dataset which is called cross-database which consisted of an entirely different population.

Main Results

In the cross-validation study, the values of accuracy, sensitivity, specificity, F1-score, and positive predictive value were approximately 0.97, 0.99, 0.95, 0.97, and 0.95, respectively; the same metrics for the cross-database test set were approximately 0.95, 0.93, 0.97, 0.96, and 0.92, respectively.

The best-performing classifier was random forest which was trained on median averaged features over 10-second segments with features from both SCG and GCG modalities.

It is worth mentioning that no hyper-parameter tuning of any kind was performed for the classifiers. Hence, there may still be room for improvement in the presented results.

Significance

The introduced machine learning pipeline could effectively extract the relevant knowledge needed for the classification of AFib using SCG and GCG data collected by smartphones. In addition, the results encouraged further investigation of SCG and GCG measurements for the detection of AFib and even other cardiac disorders.

Author's contribution

The author assisted the main author of the study with the machine learning pipeline design, verification of the overall approach and the results, and manuscript writing.

5.2 Paper II: Reliability of self-applied smartphone mechanocardiography for atrial fibrillation detection

Objectives

In [94], we assessed the reliability of self-applied SCG and GCG signals in detecting AFib.

Note that in our previous contributions, we only used recordings handled by study administrators, not by study subjects. The administrators were physicians.

Approach

As part of the MODE-AF dataset measurements, we also collected a set of self-applied measurements, which were collected solely for assessing the feasibility of self-monitoring. Self-applied measurements were recorded by the users after the first successful recording was collected by the study administrator. These recordings are referred to in our reports as self-applied measurements.

We opted to use two entirely different classification approaches for the assessment of the objective. One was based on knowledge-driven (rule-based) classification and another one was based on supervised machine learning classification.

The rule-based classification technique worked largely based on the quantification of the regularity of the signal morphology. That is, searching for a dominant frequency content that signifies the presence of a regular beating pattern.

For the supervised classification, We reused the same signal enhancement and feature extraction pipeline as in the paper described in 5.1. We utilized four different classifiers consuming the features and providing the predictions. The four classifiers were namely, random forest, SVM, XGBoost, and ANN. The knowledge-driven classification approach was previously investigated and proved effective for AFib classification in other studies of the research group.

For the evaluation of the reliability of self-applied AFib detection, we performed a subject-by-subject investigation of the predictions besides quantifying the overall performance of each classifier on the self-applied data. In this investigation, the misclassified data samples were visually inspected, argued, and reported.

Main Results

The knowledge-driven approach predicted AFib with sensitivity values of 0.96 and 0.98, specificity values of 0.98 and 0.93, and F1-score values of 0.97 and 0.95 for the physician- and self-applied measurements, respectively. Similarly, the best-performing machine learning classifier according to the F1-score, delivered, on average, sensitivity values of 0.98 and 0.94, specificity values of 0.96 and 0.94, and

F1-score values of 0.97 and 0.94, respectively.

Significance

The self-applied SCG and GCG measurements can potentially be used for differentiating AFib from SR. This new technology can help screen patients with episodic or undiagnosed AFib and also be used as a home-based self-monitoring technique.

The performance figures of almost all the tested techniques were close proving the reliability of the employed signal enhancement, feature extraction, and classification.

Author's contribution

The author was the main author of the study who designed the machine learning pipeline, performed all the machine learning-related experiments, contributed to the verification of the results, and lastly contributed to the writing and preparation of the manuscript.

5.3 Paper III: Classification of Atrial Fibrillation and Acute Decompensated Heart Failure Using Smartphone Mechanocardiography: A Multilabel Learning Approach

Objectives

In [46], our main objective was the evaluation of the feasibility of the detection of AFib and ADHF from a single joint SCG and GCG measurement. We were interested in knowing whether ADHF can be detected in the same way as AFib.

Approach

We implemented (1) a set of carefully studied and engineered features for the characterization of the signals and (2) two separate machine learning pipelines, one for multilabel and another for hierarchical classification of the two target cardiac diseases. In the hierarchical classification, first AFib was classified and the corresponding predictions were passed to the ADHF classifier besides the other features. It is worth noting that in the hierarchical classification, the input sample is limited to the subjects who have been classified into AFib. This translates to a sample population who were expected to all have AFib.

Our study was done on the MODE-AF dataset. For the evaluation, we used nested cross-validation in which we repeatedly left out one subject's data from the prepared dataset. Then, we performed hyper-parameter tuning, training, and validation utilizing K-fold cross-validation on the remaining data. Next, we predicted the test subject's classes, stored them for later post-processing, put back the left-out subject's data into the dataset, and lastly repeated the same procedure for all the subjects in the dataset.

Main Results

In the multilabel classification approach, the highest performance levels of classification, as measured with sensitivity, were 1.00 and 0.68 for AFib and ADHF, respectively. The positive predictive and negative predictive values were 0.92 and 1.00 for AFib and 0.55 and 0.88 for ADHF, respectively. The aforementioned best performances were obtained by Random Forest for AFib and Logistic Regression for ADHF.

In the hierarchical classification, the same performance levels were observed for AFib as the pipeline and the input dataset were both identical to that of the multilabel approach. For ADHF, the best-observed sensitivity was 0.68 again obtained with the logistic regression. The corresponding positive and negative predictive values were 0.70 and 0.69, respectively.

Significance

Analyzing AFib and ADHF presence from a single measurement is valuable from many perspectives. In this study, we showed that using a joint SCG and GCG and a sample size of only a limited number of ADHF patients, we could attain a relatively moderate performance for ADHF classification. The results of this study suggested that we need to implement different measures and analysis strategies for ADHF. We know that the clinical diagnosis of ADHF requires the examination of several hemodynamic parameters before and after a cardiopulmonary exercise test. In our data collection, we included no such exercise test or any direct hemodynamic parameter acquisition.

Author's contribution

The author contributed to this study together with the second author equally. The author was in charge of designing machine learning pipelines and all the respective experiments with different classifiers within multilabel and hierarchical classification frameworks. Moreover, the author contributed to the verification of the results as well as preparing and writing the manuscript.

5.4 Paper IV: sensor fusion and classification of atrial fibrillation using deep neural networks and smartphone mechanocardiography

Objectives

In [95], our main objective was to assess the feasibility of automated end-to-end feature learning and classification of AFib, SR, and Noise classes using SCG and GCG signals via deep neural networks.

Approach

We implemented (1) rotational data augmentation motivated by algebraic vector rotation, (2) a fully automated machine learning pipeline that constitutes an attention-powered deep convolutional recurrent neural network model, (3) a learnable neural network-integrated sensor fusion, (4) automated spatiotemporal feature learning through deep convolutional-recurrent blocks.

Furthermore, We assessed the usefulness and contribution of the building blocks of the deep neural network model through an ablation study. The stability and speed of convergence of the presented neural network architecture were investigated and reported as well.

Our study was done on an extended version of the MODE-AF dataset which included both physician-applied and self-applied measurements. For the evaluation, we used fully disjoint training, validation, and test sets which were created by splitting the dataset subject-wise. In the pre-processing, we performed a light noise removal and segmentation of the original records. The output of the segmentation was 10-second-long excerpts of multi-channel SCG and GCG signals. Next, the created segments were enhanced and expanded with channel-by-channel signal envelopes that were concatenated with the filtered input channels. Lastly, the rotational data augmentation was applied to the input segments to expand the size of the training and validation sets.

We reported the performance metrics on the segment and measurement levels. The measurement-level performance metrics were calculated by taking a majority vote on the predicted class labels of the respective constituting segments. We performed several experiments and repeated each experiment for ten iterations to obtain a statistical distribution over all the computed metrics and quantities we reported. We quantified the classification performance using micro-averaged and macro-averaged F1-score.

Main Results

Using the unseen test set, for segment-wise classification, micro- and macro-F1-score of 0.88 (0.87–0.89; 95% CI) and 0.83 (0.83–0.84; 95% CI) were produced, respectively. Similarly, for the measurement-wise classification micro- and macro-F1-score of 0.95 (0.94–0.96; 95% CI) and 0.95 (0.94–0.96; 95% CI) were obtained, respectively.

Significance

Our study illustrated a machine learning pipeline designed for the classification of multidimensional SCG and GCG signals with minimal signal enhancement and feature engineering efforts. We integrated many different components of conventional machine learning pipelines into a single modern deep neural network architecture. We tested the utility of the created components and proved the effectiveness of each.

Author's contribution

In this study, the author and the second author each contributed equally. Together with the second author, the author was in charge of designing the machine learning pipeline, deep neural network architecture, and all the respective experiments. Furthermore, the author contributed to the verification of the results as well as preparing and writing the manuscript.

6 Discussion and Conclusions

Remote and personalized health monitoring has many potential benefits. Early detection and progression of diseases as well as evaluation of the efficacy of interventions could be substantially improved by remote health monitoring. With the aid of remote monitoring systems, we can detect potentially harmful symptoms or risks before they become chronic or pathological. With the growing number of mobile and wearable devices, remote health monitoring is now more feasible than ever before. Wearable sensors and smartphones carry a wealth of reliable sensors that are capable of measuring many different physiological and health-related variables. Similarly, remote cardiac monitoring can have many benefits and may even be life-saving in some cases. Undiagnosed AFib, for instance, may cause irreversible and deadly complications. Using remote cardiac monitoring, other alarming heart conditions can also be detected. Consequently, the burden of cardiovascular diseases can be reduced.

In addition to accurate and precise sensing hardware, a successful remote cardiac monitoring system depends on analysis algorithms. Among all the choices of analysis algorithms, machine learning is one of the most powerful ones and, therefore, has been selected and investigated throughout this thesis work. Machine learning algorithms can uncover patterns, relationships, and insights from large and complex datasets that might be difficult or impossible to discern using manual analysis or traditional methods. Moreover, machine learning algorithms can capture nonlinear relationships between variables, allowing them to model complex systems that traditional linear methods might struggle with. These are just a few of the strengths of the machine learning algorithms that set them apart from alternative solutions.

ECG is the most widely validated and recommended measurement for detecting heart arrhythmias inside and outside clinical settings. In addition to ECG or in the absence of ECG, SCG and GCG signals can be obtained by simply logging the values of the accelerometer and gyroscope sensors of a smartphone which is placed on the chest of an individual. Through SCG and GCG we can detect AFib as long as we have proper sensor placement on the chest. The subtle changes in cardiac electrophysiology such as P wave changes may not be directly detectable by SCG and GCG signals. Therefore, we may not expect to obtain the same information and analysis results with SCG and GCG signals compared to ECG.

In Papers I, II, III, and IV, we explored the problem of AFib detection via a smartphone device as the data logger. We examined two scenarios, in one of which

a physician handled the data collection while subjects remained in a supine position. In the second scenario, the patients themselves carried out the whole process of measurement initiation, collection, and termination. Despite a slight performance decrease in the latter scenario, AFib detection was still feasible via SCG and GCG signals. The investigations were carried out via different data analysis pipelines in each of the papers.

Given that one of the main characteristics of AFib is irregularly irregular rhythm, any device or sensor that can reliably detect rhythm changes can become a viable solution for AFib monitoring. SCG and GCG signals are promising for the detection of rhythm changes and, therefore, a viable option for AFib monitoring. However, it is important to acknowledge the potential for misdiagnosing other disorders exhibiting some form of rhythm irregularity when relying solely on rhythm changes. Some such disorders are Atrial Flutter, Ectopic Atrial Rhythm, Multifocal Atrial Tachycardia, Wandering Atrial Pacemaker, as well as frequent premature atrial or ventricular contractions. This limitation needs to be further studied. However, if the target population we are screening for is suspected to have AFib and we are aiming to only measure the frequency and the duration of AFib episodes, this limitation becomes less of a concern.

As a result of the first three studies and the exploratory data analysis we have done on the MODE-AF dataset [25; 19], we understood that the rhythm regularity analysis could be simplified. The main simplification could be accomplished by augmenting the input signals with their signal envelopes. Then, limit the set of hand-crafted features to the ones that are sensitive to the regularity of the peak locations. In paper IV, we experimented with such an input set and automated feature extraction using a deep convolutional-recurrent neural network for AFib detection and obtained fairly good results. Hence, a simple peak detection and rhythm analysis could provide a moderate performance for AFib versus Sinus Rhythm classification. Nonetheless, when attempting to expand the target classes, for example by including the Noise class, simple peak detection and rhythm analysis may not be sufficiently powerful.

Throughout this doctoral work, we did not attempt to directly investigate the subtle morphological changes of the SCG and GCG signals when doing AFib or ADHF classification. But rather we focused on creating better hand-crafted features that could capture all the rhythm and morphological changes. There were two reasons for not doing an in-depth morphology analysis. Firstly, the data we gathered came with variable noise and quality levels. This was due to the fact that the population over which the MODE-AF dataset was gathered were all elderly adults who were admitted to the hospital and the technology was just introduced to the subjects briefly. The quality of the recordings can be improved by making these recordings in a more relaxed environment after receiving sufficient instructions and trial and error. Again, here we have to keep in mind the target use case and the population who is gaining the

most benefit from this technology. Secondly, based on some earlier studies on AFib detection, we knew that rhythm regularity analysis alone could deliver sufficiently high performance.

For the ADHF classification, we could not achieve the same performance levels as in the AFib classification. The low performance levels were mainly due to our ambitious objective which was the detection of ADHF using a single measurement. As it was proved in our study, the detection of ADHF is extremely challenging if we do not perform any cardiopulmonary exercise test by which we can quantify the variations in hemodynamic parameters such as pulmonary capillary wedge pressure, oxygen uptake, cardiac output, contractility, and blood pressure [96]. Without having access to the variations of the hemodynamic parameters measured before and after physically demanding tasks, the detection accuracy of ADHF likely remains low.

6.1 Potential significance

The results presented in this thesis highlight the promising potential of smartphone-based detection and monitoring of cardiovascular diseases, relying on built-in inertial sensors. These solutions offer ease of distribution and scalability, requiring only analysis software without additional hardware. The achieved accuracy and performance metrics suggest that recording durations as short as 10 to 60 seconds could yield clinically relevant results. While self-taken measurements demonstrated slightly lower performance compared to measurements taken by healthcare professionals, the difference was not substantial, indicating that with improved instructions and user practice, high-quality data can be obtained for reliable results, enabling various telehealthcare applications.

Regarding AFib detection, the combined use of SCG and GCG can serve as an alternative to photoplethysmogram (PPG) and single-lead ECG, providing comparable accuracy and potentially higher success rates than PPG without the additional hardware burden associated with ECG. However, the ADHF results did not match the reported state-of-the-art achievements using ECG or PPG, considering the limited availability of ADHF data coexisting with arrhythmia. This raises concerns about suboptimal feature selection due to the interference of arrhythmia in the learning process. It should be noted that previous ADHF studies had small sample sizes ($<$ 100 patients), which raises concerns about overfitting and generalizability. Overall, it must be emphasized that SCG and GCG should not be regarded as a replacement for clinical ECG measurements. Moreover, it is not a replacement for long-term ambulatory Holter ECG recordings. There is still a need for more research and more contemporary devices to prove the real potential of SCG and GCG for clinical and long-term ambulatory cardiac monitoring.

This doctoral thesis addresses the design, implementation, and examination of different feature engineering and machine learning pipelines for achieving the stated

objectives. Extensive exploration of various hand-crafted features targeting specific signal characteristics was conducted. The utility of different supervised classification techniques was assessed, providing a comprehensive understanding of their advantages and disadvantages. Ultimately, a complete data analysis pipeline utilizing deep neural networks for automated AFib detection was implemented and evaluated. The potential benefits of such approaches for SCG and GCG analysis are demonstrated.

6.2 Challenges

Machine learning scenarios are inherently prone to biases when attempting to distinguish between diseased populations and controls. Biases may arise due to population differences, such as the diseased population being sourced from one site while the control population is obtained from another. Such biases can give rise to situations where the model successfully separates the desired groups, but the model's parameters may inadvertently capture unintended factors, rendering the model ineffective when applied to different study settings. Moreover, the generalizability of supervised learning models often necessitates a substantial amount of data, potentially involving hundreds of patients, depending on the complexity of the disease and the technology's capability to capture disease-related features. Addressing these factors and ensuring an adequate sample size requires extensive efforts and financial resources, presenting limitations in typical academic research endeavors, including the studies presented in this thesis. Nonetheless, measures were implemented to mitigate limitations stemming from these sources, such as the careful selection of classification algorithms suitable for specific study population sizes and the application of cross-validation methods.

6.3 Future Work

The collection and proper labeling of data is a major challenge in medical science research. The dataset that was used throughout this thesis work was quite small as it contained short measurements from less than a thousand individuals. Rather than collecting larger datasets, in the future, we can focus on unsupervised representation learning (URL), which only needs unlabeled data. There has been a considerable amount of research around URL at the core of which deep neural networks were utilized.

In a URL task, the learning objective is engineered in such a way that the machine learning model is forced to learn useful and sometimes meaningful representations of the data. Once these representations are learned, the machine learning models are transferred to a downstream fine-tuning task where the representations are fine-tuned for a specific classification or regression task. Sometimes URL can be done jointly with supervised classification. In this case, the machine learning model is trained

in a semi-supervised fashion with two or more objective functions concurrently contributing to the learning process.

Among the most popular URL and semi-supervised learning techniques are,

- input reconstraction [97], denoising [98], or sparse coding via an Autoencoder [99]
- semi-supervised generative modeling via Generative Adversarial Networks (GAN) [100], semi-supervised Variational Autoencoder (VAE) [101], and semi-supervised Normalizing Flows (NF) [102]
- self-supervised representation learning via Contrastive Predictive Coding (CPC) [103], self-supervised contrastive learning [104], or self-supervised learning through auxiliary or pseudo-labeling [105]

Besides the URL mentioned above and the semi-supervised learning strategies, learning to structure the data based on the characteristics of representations of each input data point can be effective in establishing a more generalized model and/or creating models that require fewer labeled samples. Neural Graph Machines [106] and similar techniques have been recently introduced and have proven effective for this purpose.

Last but not least, in the world of medical sciences it is crucial to be able to interpret and identify the factors causing a pathological condition. When sophisticated machine learning models are utilized as a decision support system, engineers are unable to interpret and explain the reasons behind the algorithms' outputs. In addition, we know that in many cases correlation does not imply causation. Many of the popular machine learning models are not forced to look for causation and not spurious correlations. Causal machine learning has been introduced to overcome this issue and has been applied to medical data analytics [107]. Applying causal machine learning to cardiac signal analysis is one of the main themes of the author's future studies.

List of References

[1] World Health Organization. Cardiovascular diseases (cvds). https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds), June 2021. (Accessed on 06/10/2021).

[2] European Society of Cardiology. Cvd in europe and esc congress figures. https://www.escardio.org/The-ESC/Press-Office/Fact-sheets, January 2020. (Accessed on 06/10/2021).

[3] Adam Timmis, Nick Townsend, Chris P Gale, Aleksandra Torbica, Maddalena Lettino, Steffen E Petersen, Elias A Mossialos, Aldo P Maggioni, Dzianis Kazakiewicz, Heidi T May, et al. European society of cardiology: cardiovascular disease statistics 2019. *European heart journal*, 41 (1):12–85, 2020.

[4] Gerhard Hindricks, Tatjana Potpara, Nikolaos Dagres, Elena Arbelo, Jeroen J Bax, Carina Blomström-Lundqvist, Giuseppe Boriani, Manuel Castella, Gheorghe-Andrei Dan, Polychronis E Dilaveris, et al. 2020 esc guidelines for the diagnosis and management of atrial fibrillation developed in collaboration with the european association for cardio-thoracic surgery (eacts) the task force for the diagnosis and management of atrial fibrillation of the european society of cardiology (esc) developed with the special contribution of the european heart rhythm association (ehra) of the esc. *European heart journal*, 42(5):373–498, 2021.

[5] Massimo F Piepoli, Arno W Hoes, Stefan Agewall, Christian Albus, Carlos Brotons, Alberico L Catapano, Marie-Therese Cooney, Ugo Corrà, Bernard Cosyns, Christi Deaton, et al. Guidelines: Editor's choice: 2016 european guidelines on cardiovascular disease prevention in clinical practice: The sixth joint task force of the european society of cardiology and other societies on cardiovascular disease prevention in clinical practice (constituted by representatives of 10 societies and by invited experts) developed with the special contribution of the european association for cardiovascular prevention & rehabilitation (eacpr). *European heart journal*, 37(29):2315, 2016.

[6] Xinchan Jiang, Wai-Kit Ming, Joyce HS You, et al. The cost-effectiveness of digital health interventions on the management of cardiovascular diseases: systematic review. *Journal of medical Internet research*, 21(6):e13166, 2019.

[7] Sheikh Mohammed Shariful Islam and Ralph Maddison. Digital health approaches for cardiovascular diseases prevention and management: lessons from preliminary studies. *Mhealth*, 7, 2021.

[8] Numan Khan, Francoise A Marvel, Jane Wang, and Seth S Martin. Digital health technologies to promote lifestyle change and adherence. *Current treatment options in cardiovascular medicine*, 19(8):1–12, 2017.

[9] Candace Imison, Sophie Castle-Clarke, Robert Watson, and Nigel Edwards. *Delivering the benefits of digital health care*. Nuffield Trust London, 2016.

[10] Prem Prakash Jayaraman, Abdur Rahim Mohammad Forkan, Ahsan Morshed, Pari Delir Haghighi, and Yong-Bin Kang. Healthcare 4.0: A review of frontiers in digital health. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(2):e1350, 2020.

[11] Ali Yazdanyar and Anne B Newman. The burden of cardiovascular disease in the elderly: morbidity, mortality, and costs. *Clinics in geriatric medicine*, 25(4):563, 2009.

[12] John Bostrom, Greg Sweeney, Jonathan Whiteson, and John A Dodson. Mobile health and cardiac rehabilitation in older adults. *Clinical cardiology*, 43(2):118–126, 2020.

[13] Awni Y Hannun, Pranav Rajpurkar, Masoumeh Haghpanahi, Geoffrey H Tison, Codie Bourn, Mintu P Turakhia, and Andrew Y Ng. Cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network. *Nature medicine*, 25(1):65–69, 2019.

[14] Gopi Battineni, Getu Gamo Sagaro, Nalini Chintalapudi, and Francesco Amenta. The benefits of telemedicine in personalized prevention of cardiovascular diseases (cvd): a systematic review. *Journal of Personalized Medicine*, 11(7):658, 2021.

[15] Furrukh Sana, Eric M Isselbacher, Jagmeet P Singh, E Kevin Heist, Bhupesh Pathik, and Antonis A Armoundas. Wearable devices for ambulatory cardiac monitoring: Jacc state-of-the-art review. *Journal of the American College of Cardiology*, 75(13):1582–1592, 2020.

[16] Amirtahà Taebi, Brian E Solar, Andrew J Bomar, Richard H Sandler, and Hansen A Mansy. Recent advances in seismocardiography. *Vibration*, 2(1):64–86, 2019.

[17] Deepak Rai, Hiren Kumar Thakkar, Shyam Singh Rajput, Jose Santamaria, Chintan Bhatt, and Francisco Roca. A comprehensive review on seismocardiogram: Current advancements on acquisition, annotation, and applications. *Mathematics*, 9(18):2243, 2021.

[18] Szymon Sieciński, Paweł S Kostka, and Ewaryst J Tkacz. Gyrocardiography: A review of the definition, history, waveform description, and applications. *Sensors*, 20(22):6675, 2020.

[19] Jussi Jaakkola, Samuli Jaakkola, Olli Lahdenoja, Tero Hurnanen, Tero Koivisto, Mikko Pänkäälä, Timo Knuutila, Tuomas O Kiviniemi, Tuija Vasankari, and KE Juhani Airaksinen. Mobile phone detection of atrial fibrillation with mechanocardiography: the mode-af study (mobile phone detection of atrial fibrillation). *Circulation*, 137(14):1524–1527, 2018.

[20] Mohammad Saeid Mahdavinejad, Mohammadreza Rezvan, Mohammadamin Barekatain, Peyman Adibi, Payam Barnaghi, and Amit P Sheth. Machine learning for internet of things data analysis: A survey. *Digital Communications and Networks*, 4(3):161–175, 2018.

[21] Aditi Site, Jari Nurmi, and Elena Simona Lohan. Systematic review on machine-learning algorithms used in wearable-based ehealth data analysis. *IEEE Access*, 2021.

[22] Karim Bayoumy, Mohammed Gaber, Abdallah Elshafeey, Omar Mhaimeed, Elizabeth H Dineen, Francoise A Marvel, Seth S Martin, Evan D Muse, Mintu P Turakhia, Khaldoun G Tarakji, et al. Smart wearable devices in cardiovascular care: where we are and how to move forward. *Nature Reviews Cardiology*, pages 1–19, 2021.

[23] Fatma Murat, Ferhat Sadak, Ozal Yildirim, Muhammed Talo, Ender Murat, Murat Karabatak, Yakup Demir, Ru-San Tan, and U Rajendra Acharya. Review of deep learning-based atrial fibrillation detection studies. *International journal of environmental research and public health*, 18(21):11302, 2021.

[24] Theresa A McDonagh, Marco Metra, Marianna Adamo, Roy S Gardner, Andreas Baumbach, Michael Böhm, Haran Burri, Javed Butler, Jelena Čelutkienė, Ovidiu Chioncel, et al. 2021 esc guidelines for the diagnosis and treatment of acute and chronic heart failure: Developed by the task force for the diagnosis and treatment of acute and chronic heart failure of the european society of cardiology (esc) with the special contribution of the heart failure association (hfa) of the esc. *European heart journal*, 42(36):3599–3726, 2021.

[25] Mojtaba Jafari Tadi, Saeed Mehrang, Matti Kaisti, Olli Lahdenoja, Tero Hurnanen, Jussi Jaakkola, Samuli Jaakkola, Tuija Vasankari, Tuomas Kiviniemi, Juhani Airaksinen, et al. Comprehensive analysis of cardiogenic vibrations for automated detection of atrial fibrillation using smartphone mechanocardiograms. *IEEE Sensors Journal*, 19(6):2230–2242, 2018. ©2018 IEEE. Reprinted, with permission.

[26] Elizabeth A Woodcock and Scot J Matkovich. Cardiomyocytes structure, function and associated pathologies. *The international journal of biochemistry & cell biology*, 37(9):1746–1751, 2005.

[27] Paul A Iaizzo. *Handbook of cardiac anatomy, physiology, and devices*. Springer Science & Business Media, 2010.

[28] Malcolm S Thaler. *The only EKG book you'll ever need*. Lippincott Williams & Wilkins, 2021.

[29] Liang-Han Ling, Peter M Kistler, Jonathan M Kalman, Richard J Schilling, and Ross J Hunter. Comorbidity of atrial fibrillation and heart failure. *Nature Reviews Cardiology*, 13(3):131–147, 2016.

[30] T Millane, G Jackson, CR Gibbs, and GYH Lip. Acute and chronic management strategies. *Bmj*, 320(7234):559–562, 2000.

[31] Susan M Joseph, Ari M Cedars, Gregory A Ewald, Edward M Geltman, and Douglas L Mann. Acute decompensated heart failure: contemporary medical management. *Texas Heart Institute Journal*, 36(6):510, 2009.

[32] Philip B Adamson, Greg Ginn, Stefan D Anker, Robert C Bourge, and William T Abraham. Remote haemodynamic-guided care for patients with chronic heart failure: a meta-analysis of completed trials. *European Journal of Heart Failure*, 19(3):426–433, 2017.

[33] Dipak Kotecha, Carolyn SP Lam, Dirk J Van Veldhuisen, Isabelle C Van Gelder, Adriaan A Voors, and Michiel Rienstra. Heart failure with preserved ejection fraction and atrial fibrillation: vicious twins. *Journal of the American College of Cardiology*, 68(20):2217–2228, 2016.

[34] Lei Zhao, William YS Wang, and Xinchun Yang. Anticoagulation in atrial fibrillation with heart failure. *Heart Failure Reviews*, 23:563–571, 2018.

[35] Atul Verma, Jonathan M Kalman, and David J Callans. Treatment of patients with atrial fibrillation and heart failure with reduced ejection fraction. *Circulation*, 135(16):1547–1563, 2017.

[36] D George Wyse, Isabelle C Van Gelder, Patrick T Ellinor, Alan S Go, Jonathan M Kalman, Sanjiv M Narayan, Stanley Nattel, Ulrich Schotten, and Michiel Rienstra. Lone atrial fibrillation: does it exist? *Journal of the American College of Cardiology*, 63(17):1715–1723, 2014.

[37] Gregory YH Lip, Frank R Heinzel, Fiorenzo Gaita, Jose Rámon Gonzalez Juanatey, Jean Yves Le Heuzey, Tatjana Potpara, Jesper Hastrup Svendsen, Marc A Vos, Stefan D Anker, Andrew J Coats, et al. European heart rhythm association/heart failure association joint consensus document on arrhythmias in heart failure, endorsed by the heart rhythm society and the asia pacific heart rhythm society. *Ep Europace*, 18(1):12–36, 2016.

[38] Denis Roy, Mario Talajic, Stanley Nattel, D George Wyse, Paul Dorian, Kerry L Lee, Martial G Bourassa, J Malcolm O Arnold, Alfred E Buxton, A John Camm, et al. Rhythm control versus rate control for atrial fibrillation and heart failure. *New England Journal of Medicine*, 358(25): 2667–2677, 2008.

[39] Dimitrios Farmakis, Christina Chrysohoou, Gregory Giamouzis, George Giannakoulas, Michalis Hamilos, Katerina Naka, Stylianos Tzeis, Sotirios Xydonas, Apostolos Karavidas, and John Parissis. The management of atrial fibrillation in heart failure: an expert panel consensus. *Heart Failure Reviews*, 26:1345–1358, 2021.

[40] Rosita Zakeri, John M Morgan, Patrick Phillips, Sue Kitt, G Andre Ng, Janet M McComb, Simon Williams, David J Wright, Jaswinder S Gill, Alison Seed, et al. Impact of remote monitoring on clinical outcomes for patients with heart failure and atrial fibrillation: results from the rem-hf trial. *European Journal of Heart Failure*, 22(3):543–553, 2020.

[41] Omer T Inan, Pierre-Francois Migeotte, Kwang-Suk Park, Mozziyar Etemadi, Kouhyar Tavakolian, Ramon Casanella, John Zanetti, Jens Tank, Irina Funtova, G Kim Prisk, et al. Ballistocardiography and seismocardiography: A review of recent advances. *IEEE journal of biomedical and health informatics*, 19(4):1414–1427, 2014.

[42] John M Zanetti and Kouhyar Tavakolian. Seismocardiography: Past, present and future. In *2013 35th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, pages 7004–7007. IEEE, 2013.

[43] Mojtaba Jafari Tadi, Eero Lehtonen, Antti Saraste, Jarno Tuominen, Juho Koskinen, Mika Teräs, Juhani Airaksinen, Mikko Pänkäälä, and Tero Koivisto. Gyrocardiography: A new non-invasive monitoring method for the assessment of cardiac mechanics and the estimation of hemodynamic variables. *Scientific reports*, 7(1):1–11, 2017.

[44] Kouhyar Tavakolian. *Characterization and analysis of seismocardiogram for estimation of hemodynamic parameters*. PhD thesis, Applied Science: School of Engineering Science, 2010.

[45] Parastoo Dehkordi, Farzad Khosrow-Khavar, Marco Di Rienzo, Omer T Inan, Samuel E Schmidt, Andrew P Blaber, Kasper Sørensen, Johannes J Struijk, Vahid Zakeri, Prospero Lombardi, et al. Comparison of different methods for estimating cardiac timings: a comprehensive multimodal echocardiography investigation. *Frontiers in physiology*, 10:1057, 2019.

[46] Saeed Mehrang, Olli Lahdenoja, Matti Kaisti, Mojtaba Jafari Tadi, Tero Hurnanen, Antti Airola, Timo Knuutila, Jussi Jaakkola, Samuli Jaakkola, Tuija Vasankari, et al. Classification of atrial fibrillation and acute decompensated heart failure using smartphone mechanocardiography: a multilabel learning approach. *IEEE Sensors Journal*, 20(14):7957–7968, 2020. ©2020 IEEE. Reprinted, with permission.

[47] Roger Narayan. *Encyclopedia of biomedical engineering*. Elsevier, 2018.

[48] Sergio Cerutti and Carlo Marchesi. *Advanced methods of biomedical signal processing*, volume 27. John Wiley & Sons, 2011.

[49] Hsun-Hsien Chang and José MF Moura. Biomedical signal processing. *Biomedical engineering and design handbook*, 2:559–579, 2010.

[50] Chenxi Yang and Negar Tavassolian. Motion artifact cancellation of seismocardiographic recording from moving subjects. *IEEE Sensors Journal*, 16(14):5702–5708, 2016.

[51] Abdul Q Javaid, Hazar Ashouri, Alexis Dorier, Mozziyar Etemadi, J Alex Heller, Shuvo Roy, and Omer T Inan. Quantifying and reducing motion artifacts in wearable seismocardiogram measurements during walking to assess left ventricular health. *IEEE Transactions on Biomedical Engineering*, 64(6):1277–1286, 2016.

[52] Amirtaha Taebi and HA Mansy. Noise cancellation from vibrocardiographic signals based on the ensemble empirical mode decomposition. *J. Appl. Biotechnol. Bioeng*, 2(2):24, 2017.

[53] Sridhar Krishnan and Yashodhan Athavale. Trends in biomedical signal feature extraction. *Biomedical Signal Processing and Control*, 43:41–63, 2018.

[54] Mojtaba Jafari Tadi, Eero Lehtonen, Tero Hurnanen, Juho Koskinen, Jonas Eriksson, Mikko Pänkäälä, Mika Teräs, and Tero Koivisto. A real-time approach for heart rate monitoring using a hilbert transform in seismocardiograms. *Physiological measurement*, 37(11):1885, 2016.

[55] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE mobile computing and communications review*, 5(1):3–55, 2001.

[56] Steven M Pincus. Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences*, 88(6):2297–2301, 1991.

[57] Joshua S Richman and J Randall Moorman. Physiological time-series analysis using approximate entropy and sample entropy. *American Journal of Physiology-Heart and Circulatory Physiology*, 278(6):H2039–H2049, 2000.

[58] Steven Smith. *Digital signal processing: a practical guide for engineers and scientists*. Elsevier, 2013.

[59] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[60] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

[61] Christopher M Bishop. Pattern recognition. *Machine learning*, 128(9), 2006.

[62] Stephen J Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015.

[63] Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)*, 67(2):301–320, 2005.

[64] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[65] Alex J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and computing*, 14(3):199–222, 2004.

[66] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[67] Leo Breiman, Jerome H Friedman, Richard A Olshen, and Charles J Stone. *Classification and regression trees*. Routledge, 2017.
[68] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.
[69] J Ross Quinlan. *C4. 5: programs for machine learning*. Elsevier, 2014.
[70] Wei-Yin Loh. Classification and regression trees. *Wiley interdisciplinary reviews: data mining and knowledge discovery*, 1(1):14–23, 2011.
[71] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.
[72] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. Boosting and additive trees. In *The elements of statistical learning*, pages 337–387. Springer, 2009.
[73] Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. Multi-class adaboost. *Statistics and its Interface*, 2(3):349–360, 2009.
[74] Yoav Freund. A more robust boosting algorithm. *arXiv preprint arXiv:0905.2138*, 2009.
[75] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28 (2):337–407, 2000.
[76] Llew Mason, Jonathan Baxter, Peter L Bartlett, Marcus Frean, et al. Functional gradient techniques for combining hypotheses. *Advances in Neural Information Processing Systems*, pages 221–246, 1999.
[77] Philip M Long and Rocco A Servedio. Random classification noise defeats all convex potential boosters. *Machine learning*, 78(3):287–304, 2010.
[78] Jerome H Friedman. Stochastic gradient boosting. *Computational statistics & data analysis*, 38 (4):367–378, 2002.
[79] Llew Mason, Jonathan Baxter, Peter Bartlett, and Marcus Frean. Boosting algorithms as gradient descent. *Advances in neural information processing systems*, 12, 1999.
[80] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pages 1189–1232, 2001.
[81] Ping Li. Robust logitboost and adaptive base class (abc) logitboost. *arXiv preprint arXiv:1203.3491*, 2012.
[82] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.
[83] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.
[84] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.
[85] Yann LeCun et al. Generalization and network design strategies. *Connectionism in perspective*, 19(143-155):18, 1989.
[86] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial intelligence review*, 53(8): 5455–5516, 2020.
[87] Ian J Goodfellow. Technical report: Multidimensional, downsampled convolution for autoencoders. *Universit 'e de Montr é al, inf. t 'e c*, 2010.
[88] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.
[89] Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.
[90] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, chapter Model Assessment and Selection, pages 219–259. Springer Science & Business Media, 2009.
[91] Hao Cheng, Dorian J Garrick, and Rohan L Fernando. Efficient strategies for leave-one-out cross validation for genomic best linear unbiased prediction. *Journal of animal science and biotechnology*, 8:1–5, 2017.

[92] Sudhir Varma and Richard Simon. Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, 7(1):1–8, 2006.

[93] Gavin C Cawley and Nicola LC Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *The Journal of Machine Learning Research*, 11:2079–2107, 2010.

[94] Saeed Mehrang, Mojtaba Jafari Tadi, Tero Hurnanen, Timo Knuutila, Olli Lahdenoja, Jussi Jaakkola, Samuli Jaakkola, Tuija Vasankari, Tuomas Kiviniemi, Juhani Airaksinen, et al. Reliability of self-applied smartphone mechanocardiography for atrial fibrillation detection. *IEEE Access*, 7:146801–146812, 2019. ©2019 IEEE. Reprinted, with permission.

[95] Saeed Mehrang, Mojtaba Jafari Tadi, Timo Knuutila, Jussi Jaakkola, Samuli Jaakkola, Tuomas Kiviniemi, Tuija Vasankari, Juhani Airaksinen, Tero Koivisto, and Mikko Pänkäälä. End-to-end sensor fusion and classification of atrial fibrillation using deep neural networks and smartphone mechanocardiography. *Physiological Measurement*, 43(5):055004, 2022. ©2022 Physiological Measurement. Reprinted, with permission.

[96] Luke Neill, Mozziyar Etemadi, Liviu Klein, and Omer T Inan. Novel noninvasive biosensors and artificial intelligence for optimized heart failure management. *Basic to Translational Science*, 7 (3_Part_2):316–318, 2022.

[97] Anupriya Gogna, Angshul Majumdar, and Rabab Ward. Semi-supervised stacked label consistent autoencoder for reconstruction and analysis of biomedical signals. *IEEE Transactions on Biomedical Engineering*, 64(9):2196–2205, 2016.

[98] Mohamad Mahmoud Al Rahhal, Yakoub Bazi, Haikel AlHichri, Naif Alajlan, Farid Melgani, and Ronald R Yager. Deep learning approach for active classification of electrocardiogram signals. *Information Sciences*, 345:340–354, 2016.

[99] Alireza Makhzani and Brendan Frey. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

[100] Augustus Odena. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583*, 2016.

[101] Jong-Hwan Jang, Tae Young Kim, Hong-Seok Lim, and Dukyong Yoon. Unsupervised feature learning for electrocardiogram data using the convolutional variational autoencoder. *PloS one*, 16(12):e0260612, 2021.

[102] Pavel Izmailov, Polina Kirichenko, Marc Finzi, and Andrew Gordon Wilson. Semi-supervised learning with normalizing flows. In *International Conference on Machine Learning*, pages 4615–4630. PMLR, 2020.

[103] Temesgen Mehari and Nils Strodthoff. Self-supervised representation learning from 12-lead ecg data. *Computers in Biology and Medicine*, 141:105114, 2022.

[104] Dani Kiyasseh, Tingting Zhu, and David A Clifton. Clocs: Contrastive learning of cardiac signals across space, time, and patients. In *International Conference on Machine Learning*, pages 5606–5615. PMLR, 2021.

[105] Pritam Sarkar and Ali Etemad. Self-supervised learning for ecg-based emotion recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3217–3221. IEEE, 2020.

[106] Thang D Bui, Sujith Ravi, and Vivek Ramavajjala. Neural graph learning: Training neural networks using graphs. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 64–71, 2018.

[107] Jonathan G Richens, Ciarán M Lee, and Saurabh Johri. Improving the accuracy of medical diagnosis with causal machine learning. *Nature communications*, 11(1):1–9, 2020.

TURUN
YLIOPISTO
UNIVERSITY
OF TURKU

Painosalama, Turku, Finland 2023

ISBN 978-951-29-9542-4 (PRINT)
ISBN 978-951-29-9543-1 (PDF)
ISSN 2736-9390 (Painettu/Print)
ISSN 2736-9684 (Sähköinen/Online)